Office办公应用非常之旅

热门领域的实用VBA系统，办公领域开发者的珍贵资料！

一线开发者倾力打造，深入剖析VBA开发中的重点和难点！

Excel

VBA应用开发经典案例

姚文涛 编著



- 通过9个典型应用案例详细讲解Excel VBA的各种常见应用
- 使用最新的2007版本，讲解最新的界面和技术
- 范围涉及销售管理、人事管理、成绩管理和办公管理等多个领域
- 流程清晰，代码详细，步骤完整，让读者以最直观的方法进行学习
- 涵盖了Excel对象、ADO和SQL语句等多项VBA核心技术
- 适用于Excel 2002/2003/2007等多个版本
- 提供QQ群即时答疑和网站论坛在线答疑服务



光盘内容

- 1000个Excel常用技巧（免费赠送电子书，共326页）
- 13个Excel基础操作多媒体视频演示（免费赠送）
- 12大类共102个常用Excel模版文件（免费赠送）
- 本书涉及的18个实例的多媒体视频演示文件
- 书中所涉及的实例源文件供学习使用

清华大学出版社

office 办公应用非常之旅

Excel VBA 应用开发经典案例

姚文涛　编著

清华大学出版社

北　京

内 容 简 介

本书分为 3 篇，分别是基础篇、简单实例篇和复杂实例篇，一共包含 12 章。第 1~3 章是基础篇，介绍 Excel VBA 开发的基础知识，包括熟悉 VBE 开发环境、VBA 程序设计基础和 Excel 对象模型知识。第 4~9 章是简单实例篇，结合各个实用实例，介绍如何通过 VBA 代码调用 Excel 本身的各种强大数据管理与分析功能，内容涉及数据有效性、排序、自动筛选、高级筛选、名称、工作表函数和工作表保护等。第 10~12 章是复杂实例篇，重点讲述了 Excel 2007 结合数据库的开发模式。读者可以认识和了解 DAO/ADO/ADOX 对象，使用 SQL 查询语句。

本书从实际出发，对每个实例都介绍了设计思路与知识点，避免读者在学习过程中走弯路。无论是初学者还是有一定基础的读者，都可以通过学习本书来编写自己的应用程序。

本书适合有大量数据处理需求的管理人员阅读，也适合大中专院校学生以及电脑爱好者学习阅读。

图书在版编目（CIP）数据

Excel VBA 应用开发经典案例/姚文涛编著. —北京：清华大学出版社，2009.7
（Office 办公应用非常之旅）

ISBN 978-7-302-19759-1

I. E…　II. 姚…　III. 电子表格系统，Excel　IV. TP391.13

中国版本图书馆 CIP 数据核字（2009）第 040932 号

责任编辑：朱英彪　周中亮
封面设计：刘　超
版式设计：杨　洋
责任校对：张彩凤
责任印制：
出版发行：清华大学出版社　　**地　址：**北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　编：**100084
社 总 机：010-62770175　　**邮　购：**010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：
装 订 者：
经　　销：全国新华书店
开　　本：190×260　**印　　张：**32.25　**字　　数：**742 千字
（附光盘 1 张）
版　　次：2009 年 7 月第 1 版　**印　　次：**2009 年 7 月第 1 次印刷
印　　数：1～5000
定　　价：48.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。
联系电话：010-62770177 转 3103　　产品编号：028201-01

前　言

Preface

随着当今社会经济与信息技术的飞速发展，人们每天接触到的信息量与以往不可同日而语。无论是个人还是企业，要管理好这些信息都是一份相当重要的工作。Excel 2007 是一款很强大的数据处理和计算分析软件，并且已经为广大用户接受。笔者有理由相信，大多数读者都曾经或正在使用 Excel，并且接触到 VBA 开发功能。

本书正是基于以上两点，以 Excel 2007 为平台，通过众多实例展现 Excel 2007 强大的数据处理与分析功能，让未接触到 VBA 开发的读者步入 VBA 开发的殿堂，让已了解 VBA 开发的读者积累更多的实战经验。

本书的内容并不局限于 Excel 2007 本身的内容。在本书讲解的部分实例中将 Excel 2007 与数据库结合起来，极大地提高数据处理与分析的速度。而 Excel 2007 的工作簿文件本身也可以被认为是一个数据库，在本书中也有相应实例展现如何操作 Excel 数据库文件。

本书特点

1．循序渐进，由浅入深

本书针对广大用户的情况，采取由浅入深的方式展开。本书前 3 章是基础章节，介绍 Excel VBA 开发的基础知识，包括熟悉 VBE 开发环境、VBA 程序设计基础和 Excel 对象模型知识。在实例的安排上，前面的章节讲述的基本上是较为简单的实例，复杂的实例都放置在本书后面的章节。

2．实用实例，内容丰富

本书中所讲述的实例都具有很强的实用价值。实例用途各不相同，其开发方式也各不一样，部分实例之间的开发方式还具有较强的比较性，用户可以根据各自的优缺点采用相应的开发方式。在选材上，也不单一地选择商业应用方向的实例。

3．知识点提示，加深理解

在本书的实例章节中，基本上都会有知识点提示。这些知识点都是对应章节中使用到的难点与重点知识。了解并熟练掌握这些知识不仅便于实例理解，也可丰富读者的开发途径。安排知识点在每个章节前面便于读者及时查找疑点问题，加深理解。

4．注释详细，图文并茂

本书的程序代码都使用统一的格式标识，在代码块的每一句后面都会标识注释，对该语句完成什么工作加以说明。对于比较复杂的过程或函数，为了能够让读者更容易理解其意图，

在讲述中不仅通过文字说明其运行流程，还通过流程图展示其过程，其中的一些重点与难点都在其中加以讲述。

5．配有光盘，加速学习

本书配套光盘中包含了书中相关操作内容以及各个实例的源文件。

本书内容

第 1 章：如果读者还是一个新手，这一章将引导读者步入 VBA 开发殿堂。这部分内容包括熟悉 VBE 开发环境、调试工具和认识宏。

第 2 章：该章节介绍 VBA 程序设计基础。内容包括数据类型、常量、变量、过程与函数、表达式与运算符、结构语句和数组。

第 3 章：本章对 Excel 对象模型进行了比较简单的介绍。由于 Excel 对象模型包含的对象繁多，本章只讲述 4 个主干对象的常用属性和方法。这些主干对象是应用程序对象 Application、工作簿对象 WorkBook、工作表对象 WorkSheet 和单元格区域对象 Range。

第 4~9 章：这一部分包含了 6 个较简单的实例，其复杂程度依次呈上升趋势。这一部分的实例涉及客户管理、学生成绩管理、固定资产管理、进销存管理、人事管理和商场销售管理。涉及的知识点包括窗体的设计、工作表界面设计、数据有效性、自动筛选、单元格控制、自定义菜单、名称、工作表函数应用和加载宏等。

第 10~12 章：这一部分包含了 3 个较为复杂的实例，大部分都使用到了数据库知识。这 3 个实例分别是学生座位编排、合同管理、拆分和备份工作簿工具。涉及的知识点包括工作表可见性、保护与撤销工作表保护、DAO/ADO 数据库对象、SQL 查询、ADOX 对象。

读者对象

1．VBA 入门与有一定经验的读者；
2．公司管理人员；
3．实验室、研究所数据分析处理人员；
4．会计、审计、统计等工作人员；
5．大中专院校学生。

本书作者

本书由姚文涛统筹编写，同时王俊标、陈晨、高守传、郭瑞、周宇炜、蔡雪焘、陈杰、荣飞、郑林、张路平、项宇峰、罗皓菡、赵正坤、公芳亮、程明雷、梁文建、马斗、邱哲、宋昕、陈刚、强致懿、郭腊梅、肖萍、程鹏辉、吕静、贺广治等参与了本书的编写、整理和统稿工作，在此一并表示感谢！

编　者

目　录

Contents

第 1 篇　Excel VBA 基础知识

第 2 篇 简单实例

第 3 篇　复杂实例

第1篇

Excel VBA 基础知识



第 1 章　初识 Excel 2007 VBA

本章以及后续的两个章节是 Excel 2007 VBA 知识的奠基章节。本章主要讲解 Excel 2007 VBA 的开发环境（VBE）、调试工具和宏的使用方法。“工欲善其事，必先利其器”，认真掌握该部分内容，可以极大地提高开发者的开发效率与调试效率。

1.1　VBA 的能力

这些年笔者在所工作的每个地方几乎都会问周围经常使用 Excel 进行数据分析与管理的同事同一个问题，为什么不使用 VBA 处理那些繁琐的日常事务？得到的回答不外乎以下答案：VBA 是什么，有用吗；VBA 太难了，看了很多书也不得要领，不知道怎么把它运用到实际工作中。

笔者想，翻看本书的读者也有很大部分存在以上的疑惑。对于前者笔者想打个比方来说，聚会的时候大家都喜欢喝啤酒，可是开启啤酒瓶就“八仙过海，各显神通”了，有的练牙齿功夫，有的抓着啤酒瓶子照着菱角分明的桌子或其他硬物上敲，有的摇动啤酒瓶来个井喷，但是对于笔者而言更喜欢拿橇子来解决问题！VBA 就像开啤酒瓶的橇子。

笔者的一位同学曾经碰到一个问题：同学有十几个工作簿，每个工作簿只有一个表，每个表里有 10000 行左右，而其工作即是将里面部分数据行删除。删除的规律是：电话号码列包括座机号（带区号）或手机号，该列最多包括两个号码，如果有两个电话时，使用“/”隔开，而需要删除的是那些只有座机号的行。

在碰到该问题之前，笔者的同学并不会使用自动筛选、函数等功能，更不用说 VBA，因而只能完全手动操作。10 多万行数据，一行一行手动处理，这是很大的工作量！需要多少时间和精力处理？当同学求助于我时，我使用两个方法解决了该问题。

其一是使用自动筛选和函数。首先新建一个列，该列使用函数按照“/”分开两个电话号码。然后取得两个号码的第一位数，把它们相加。如果都是座机号，那么这个数字一定为 0。如果包含了手机号，该结果将大于 0。然后通过自动筛选，将该列为 0 的数据筛选出来后删除。

其二是写一个加载宏，在这个加载宏里面包含了一个自定义函数。函数返回的就是前面相加的结果。对于每一个工作簿将省去每次都需要手动书写公式的麻烦。两种方法大体上是一样的，区别只是前者完全是通过 Excel 2007 的界面操作完成，而后者通过编程完成，前后只花几分钟的时间。

由此可以看出，全面了解 Excel 2007 的功能和掌握一部分编程技巧，将极大地提高办公效率。

对于后一种读者，笔者想说的是，VBA 是一种很通俗易懂的计算机语言，通常所欠缺的

是发现问题、分析问题的思维过程。笔者建议在阅读本书的实例时，首先构思一下应该如何完成该系统，然后把解决方案实现出来，最后再来看笔者的方案是如何做的。

到底 VBA 能帮助使用者做哪些工作呢？其实这个问题很难界定，使用 VBA 开发应用是一份创造性的工作，在很大程度上依赖于开发者本人的创造力。但是有一点是必然的，VBA 是一种程序语言，它需要有确切的执行规律。

例如电话号码列，如果没有“/”分割两个号码或者座机号码不带区号，那么后面所使用的解决办法就不能有效执行了。往往笔者在使用 VBA 开发应用时，第一步就是找到潜在的规律，因此每一个疑难问题的前期分析问题步骤很重要。

知识点：高级筛选

在 Excel 2007 中处理数据时，经常需要按照某些条件对数据进行筛选，通常可以采用自动筛选功能实现。但是有时候筛选的条件十分复杂或者需要筛选出某列的不重复项，此时就需要高级筛选。自动筛选将在第 5 章学生成绩管理系统中介绍，读者可以通过该章的知识点了解该功能。由于本章的实例中接触到了高级筛选，因而将高级筛选的知识在此加以介绍。

高级筛选的操作既可以通过菜单实现操作也可以使用 VBA 代码实现，详细的界面操作可以参见 1.4.2 节的介绍。这里详细介绍高级筛选的代码实现及其语法格式的解释。高级筛选是通过单元格对象或区域的 AdvancedFilter 方法实现的。该方法的语法如下：

```
表达式.AdvancedFilter(Action, CriteriaRange, CopyToRange, Unique)
```

该方法的几个参数的意义如下。

- Action 参数：必选参数。用于指定是否就地复制或筛选列表。
- CriteriaRange 参数：可选参数。该参数制定条件区域，如果省略该参数，则没有条件限制。
- CopyToRange 参数：可选参数。如果 Action 为 xlFilterCopy，则为复制行的目标区域；否则，忽略该参数。
- Unique 参数：可选参数。如果为 True，则只筛选唯一记录；如果为 False，则筛选符合条件的所有记录。默认值为 False。

这里不再列出该方法的实例，读者可以参见本章最后一节中录制宏实例小节的代码。

1.2　认识 VBA 编辑器（VBE）

在正式进入 Excel VBA 开发之前，有必要了解一下 VBE 开发环境。图 1-1 显示了一个 VBE 开发环境界面，从图中可以看到 VBE 开发环境的 3 个基本组成部分：工程资源管理器、属性和程序设计窗口。要从 Excel 2007 的界面进入 VBE 可以采用两种方式：

- 按 Alt+F11 组合键。
- 将 Visual Basic 编辑器按钮设置在快速工具栏上。在 Excel 2007 的快速工具栏（如

图 1-2 所示）上单击 按钮，在随后弹出的菜单中选择【其他命令】选项（如图 1-3 所示），即会弹出自定义 Excel 选项窗口（如图 1-4 所示）。在该窗口的选择命令下拉列表框中选择【所有命令】，然后在其下方的列表框中选择【Visual Basic】选项，单击【添加】按钮。确认后在快速工具栏上将会出现快速访问 VBE 的按钮。



图 1-1　VBE 编辑环境



图 1-2　快速访问工具栏



图 1-3　快速访问工具栏菜单

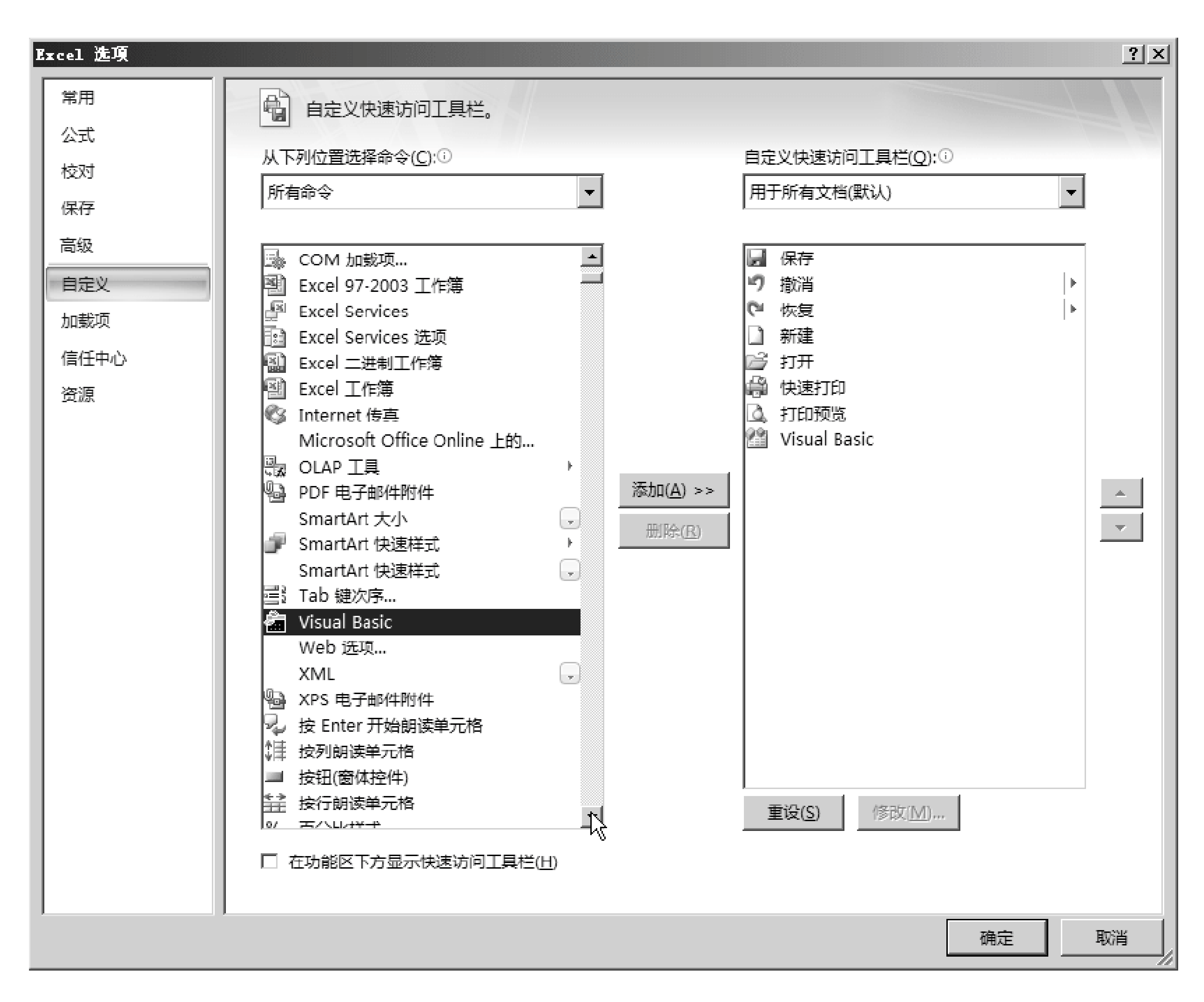


图 1-4　Excel 选项

1.2.1　VBE 环境的设置

VBE 中的环境设置可以允许用户自行定制，开发者可以按照自己的习惯设置这些选项。下面讲述几个经常使用到的较有帮助的环境设置。

依次选择【工具】|【选项】|【编辑器】命令（如图 1-5 所示），选中【要求变量声明】复选框。该复选框被选中后，编辑器会自动在每个 Microsoft Excel 对象、模块和窗体的代码段首行插入下列语句：

```
Option Explicit
```

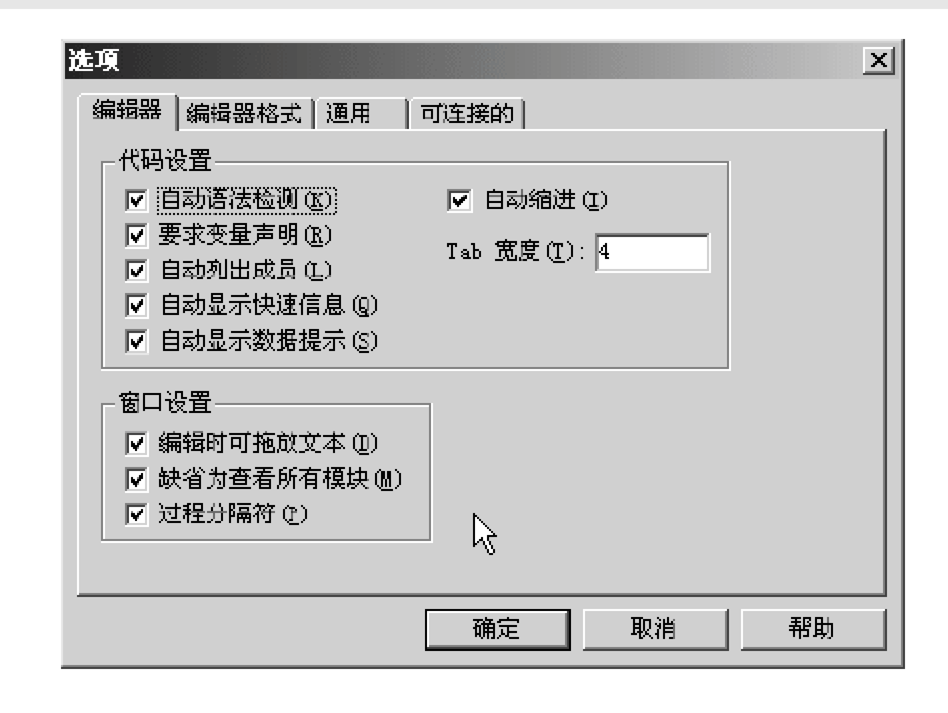


图 1-5　VBE 环境设置

当代码段首行出现该语句时，在代码中必须使用 Dim、Private、Public 或 ReDim 语句显式声明所有变量。试图使用未声明的变量名将发生编译错误。使用 Option Explicit 可避免拼错

现有变量的名称，或避免在变量范围不清楚的代码中产生混淆。

另外，在【通用】选项卡中可以设置窗体设计时网格的间距。在选中了【对齐到网格】复选框的情况下，设置一个比较小的间距单位有利于窗体的编辑操作。这个值可以设置的范围为 2~60（磅），笔者习惯设置为 2。

1.2.2 VBE 编辑器工具栏

VBE 编辑器工具栏包括常用的功能工具按钮（如图 1-6 所示）。该工具栏包含了 3 部分按钮，前面两个按钮用于切换到 Excel 2007 界面和插入用户自定义对象，这些对象包括用户窗体、模块、类模块和过程。中间部分是编辑按钮，包括保存、剪切、复制、粘贴、查找、撤销和重复。随后的一些按钮用于 VBA 程序调试及打开管理器窗口，包括运行程序、中断、重新设置、设计模式、工程资源管理器、属性窗口、对象浏览器和工具箱。

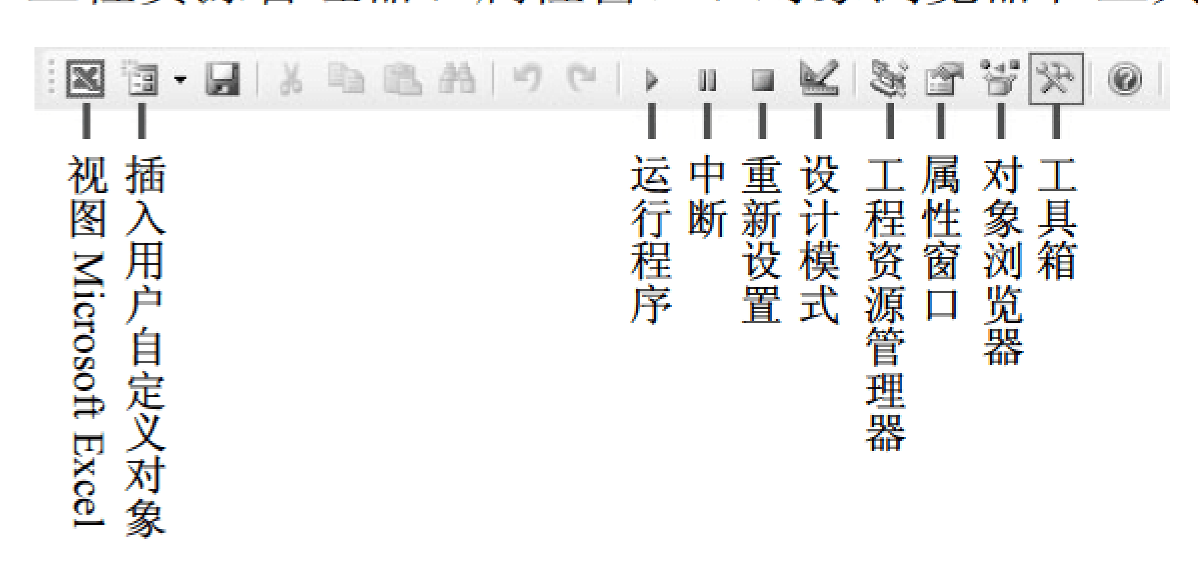


图 1-6 VBE 编辑器工具栏

1.2.3 工程资源管理器

工程资源管理器实现了对 Excel 2007 VBA 工程文件资源的统一管理。该管理器列出了所有已打开的工作簿和所加载的附加项（例如加载宏）。要显示该管理器窗口，有 3 种方法：一种是依次选择【视图】|【工程资源管理器】命令；一种是按 Ctrl+R 快捷键；一种是单击工具栏中的【工程资源管理器】按钮。对于每个 VBA 工程，都只可能包含 4 类对象：Microsoft Excel 对象、窗体、模块和类模块，工程资源管理器如图 1-7 所示。



图 1-7 工程资源管理器

- ❑ Microsoft Excel 对象：该对象下包含了 Excel 的表对象和 Thisworkbook 工作簿对象，双击某个对象，将进入该对象的代码编辑窗口。
- ❑ 窗体：用户自定义的对话框或窗体界面，用于辅助输入或显示输入。双击后将会显示窗体。如果要查看窗体代码，需要右击窗体，然后选择查看代码。
- ❑ 模块：保存自定义 VBA 代码，录制宏时，新录制的宏也将保存在该处。双击时可以直接进入代码编辑。
- ❑ 类模块：保存以类或对象方式编写的代码。通过创建

类模块，可以创建用户自定义的类和对象。使用已建立的类模块，不需要编码者了解它具体是如何工作的，因此可以实现共享代码。

Microsoft Excel 对象中包含的对象不能在 VBE 环境下添加或删除。当用户需要添加或删除其中的对象时（例如工作表），可返回 Excel 2007 操作界面操作。其他 3 种对象的添加有 3 种方式：

- 选择【插入】菜单，然后选择插入的对象类型。
- 右击工程文件名，选择【插入】选项，此时可以选择插入的对象类型。
- 在 VBE 编辑工具栏上单击第二个按钮右边的下拉箭头，选择插入对象类型。

1.2.4　属性窗口

在属性窗口中，可以查看和设置工程中不同对象的属性，包括工作表、工作簿、模块和窗体控件。在属性窗口中，对象的属性可以按照两种方式查看，分别是按字母序查看和按分类序查看（如图 1-8 所示）。只需单击属性窗口中对应的标签就可以进入相应的查看模式。



图 1-8　属性窗口

- 按字母序：按字母顺序列出被选择对象的所有属性。选择属性名，输入或选择新设置即可修改属性设置。
- 按分类序：按类别列出选中对象的所有属性。可以将清单折叠起来以便查看。不同对象展现的类别并不一致。

属性窗口除了可以修改工程、对象、模块的属性外，更多是用于对用户窗体各个对象属性的交互设计。调用属性窗口的方法如下：

- 选择【视图】菜单，然后选择【属性】命令。
- 在键盘上按 F4 快捷键。
- 从工具栏中单击【属性窗口】按钮。

1.2.5　代码窗口

在工程资源管理器的对象上双击可开启该对象的代码窗口，查看或编辑该对象的代码（如图 1-9 所示）。在这里可以完成查看修改录制的宏代码和现存的 VBA 工程代码工作。

给对象编写代码时，一般的操作是：首先从【对象】下拉列表框中选择需要编程的对象，图中选择了 Label 客户管理标签对象，然后需要选择相应的事件，VBE 将自动生成事件过程的结构代码，用户在该结构中输入代码即可。

代码窗口的显示方式有两种：全模式视图和过程视图。在代码窗口的左下方可以找到各自的激活按钮。下面具体介绍这两种显示方式。

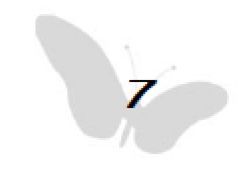




图 1-9　代码窗口

- 全模式视图：默认项，在代码窗口中显示该对象的每个子过程代码。
- 过程视图：只显示当前过程代码。

要开启某个对象的代码窗口可以采取以下 3 种方法：

- 在工程浏览器窗口中选择需要的用户窗体或者模块，然后单击【查看代码】按钮。
- 在菜单中依次选择【视图】|【代码】命令。
- 在键盘上按 F7 快捷键。

1.2.6　对象浏览器

对象浏览器（如图 1-10 所示）用于查看该工程中所有引用、控件的属性方法和事件的浏览器以及 VBA 全局常量。通过该窗口，可以方便而迅速地查找 Excel 2007 中对象所包含的类、属性、方法和事件。找到该对象后，单击它，然后按 F1 快捷键即可打开该对象属性、方法和事件的帮助文档。该窗口是一个强有力的 VBA 应用开发辅助窗口。打开该窗口的方式有两种：



图 1-10　对象浏览器

- 在菜单中依次选择【视图】|【对象浏览器】命令。
- 在 VBE 环境中，按 F2 快捷键。

1.3　VBE 调试工具

VBE 开发环境提供了几个常用的调试工具。恰当地使用这些工具，对于查看宏代码与定位问题代码段十分有用，极大地提高了开发效率。这些工具包括逐句调试、断点设置、设置下一条语句、运行到光标、立即窗口、悬浮窗口和监视窗口，以下将分别介绍。

1.3.1　逐句调试

在通常情况下，宏代码将被全部逐行执行，这个过程很快，一旦出现问题，不便于查找问题所在。而使用逐句调试，可以一次运行一句代码，便于查找问题代码行所在。在菜单中选择【调试】|【逐语句】或者按 F8 快捷键都可以进入逐语句调试模式。进入该模式后，当前被执行的代码语句将会用黄色突出显示，同时代码左侧会显示一个黄色箭头，该效果如图 1-11 所示。再次选择【逐句语句】或按 F8 快捷键，程序将执行该语句行，并跳转到下一条语句。



图 1-11　逐句调试

1.3.2　断点设置

如果运行的代码行数很多，使用逐句调试显得很繁琐。特别是问题代码行所处位置比较靠后时，一次执行很耗费时间。当知道问题行大概的所在位置时，可以设置一个断点，让程序运行到该断点时中断，然后使用前面讲的逐句调试，逐句执行语句。这样可以迅速定位问题的真实所在，提高调试代码的效率。

要设置断点，首先在需要中断的代码行左侧单击。此时，一个褐色的圆点将会显示在代码行左侧，该代码行也将会突出显示为褐色（如图 1-12 所示）。当程序运行到该行代码时，程序中止执行，该行代码将突出显示为黄色，意即该行为当前执行行。

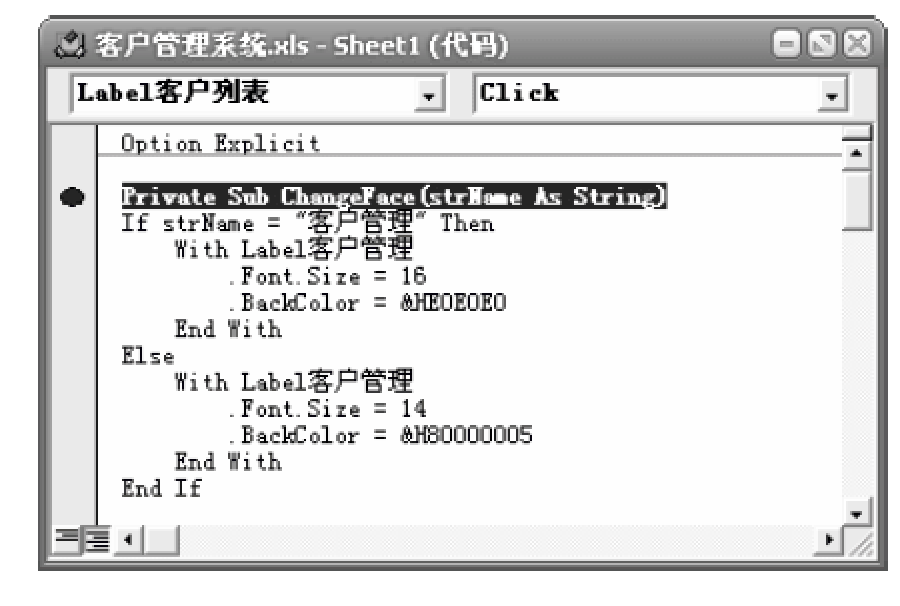


图 1-12　断点设置

1.3.3　设置下一条语句

当逐句调试代码时，可能需要跳过某些代码，或者已经修改了某些代码，现在需要重新运行时，都需要重新设置下一条运行代码的位置。可以采用的方法有以下两种：

- 使用鼠标将该黄色箭头拖动到下一条需要运行的语句的前面（如图 1-13 所示）。
- 在需要跳转到的行中单击一下，然后选择【调试】|【设置下一条语句】命令。

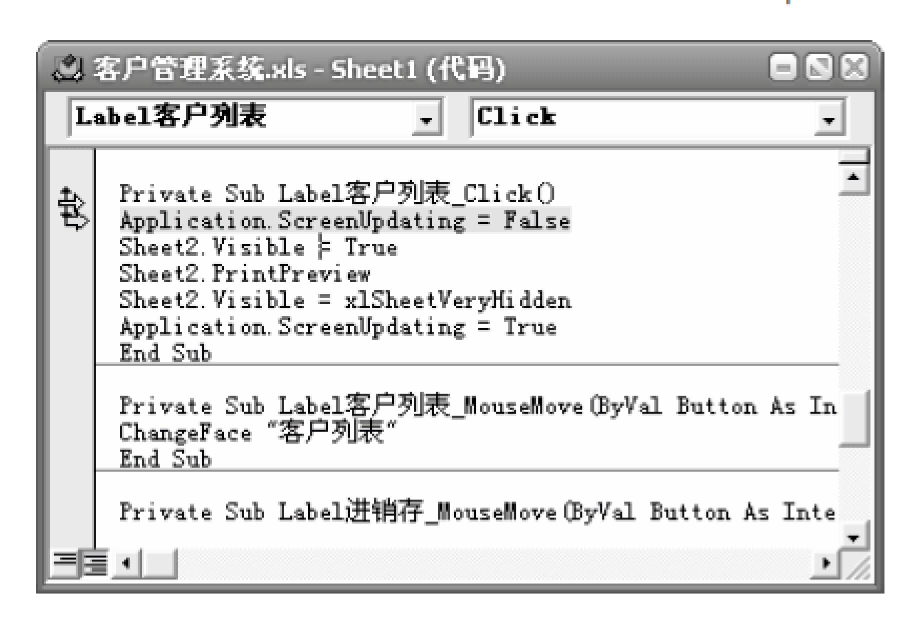


图 1-13　设置下一条语句

注意：在做该操作前，首先需要进入逐句调试模式。

1.3.4　运行到光标

当逐句调试代码时，可能需要一次性运行一段代码，而不是一步一步地执行。在调试循环结构时，这种情况经常出现。例如可能需要让某段代码循环运行 100 次，然后跳出循环，在执行随后的代码时暂停。此时可以按 Ctrl+F8 组合键或者选择【调试】|【运行到光标】命令来实现该目的。

1.3.5　立即窗口

立即窗口在调试中可以完成查询变量的工作，另外它也可以接受部分命令语句，例如 thisworkbook.save 、sheet1.select 等。按 Ctrl+G 组合键即可打开该窗口（如图 1-14 所示）。当需要查看某个变量的值时，可以使用 debug.Print 命令语句和变量名来查询。图中例子使用该语句查看了当前工作簿的名称。

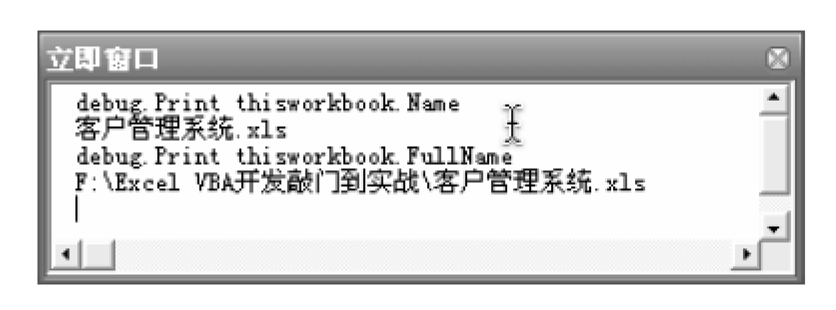


图 1-14　立即窗口

1.3.6　悬浮窗口

在调试模式下，将鼠标悬浮在代码表达式的上方，等待几秒，将会弹出一个工具提示。该工具提示显示当前表达式的值，效果如图 1-15 所示。图中显示的提示是工作表 sheet6 中 Cells(i,1)单元格的值。

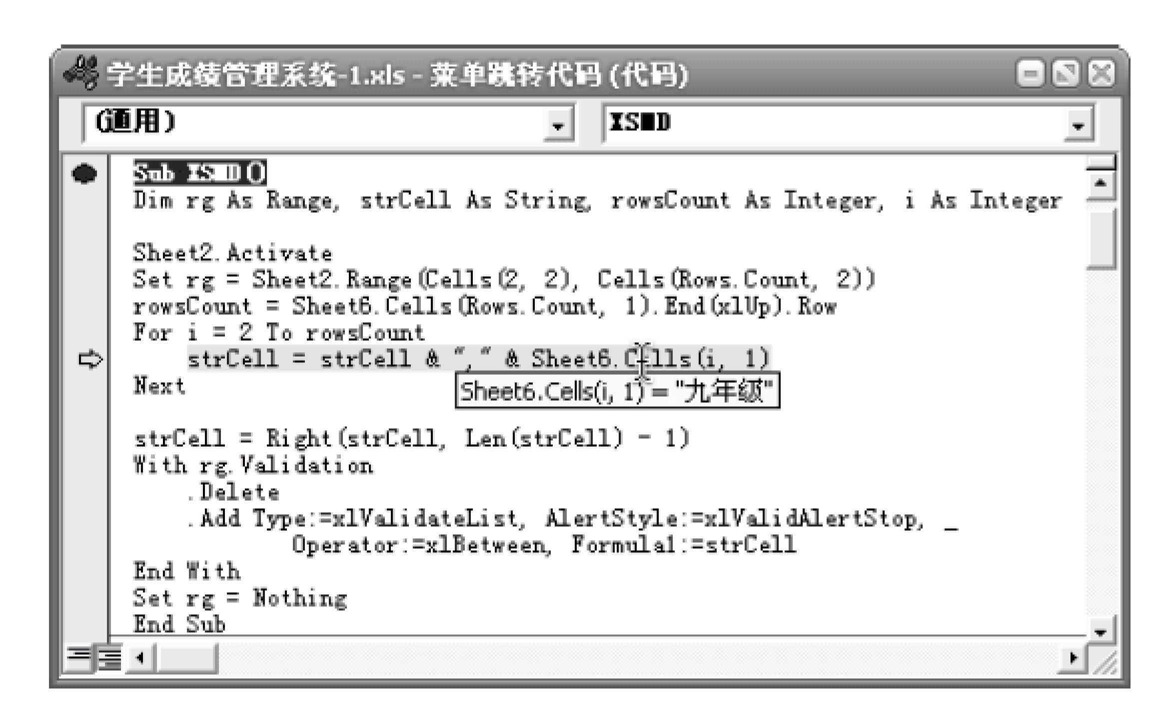


图 1-15　悬浮窗口

1.3.7　监视窗口

监视窗口允许在逐句运行代码时监视任何表达式的值，这与立即窗口和悬浮窗口的查询功能类似。当在调试中需要检查很多变量时，使用该窗口比较便利，建立监视也比较简便。如果希望在调试过程中监视选中单元格的内容，即 Selection.value，操作步骤如下：

（1）在 VBE 菜单中依次选择【调试】|【添加监视】命令（如图 1-16 所示）。

（2）在【添加监视】对话框中的【表达式】文本框中输入 Selection.value。

（3）在【监视类型】选项组中选中【监视表达式】单选按钮，然后单击【确定】按钮。

此时，在监视窗口中即会显示所添加的监视表达式 Selection.value，以及相应的当前值、类型等信息，如图 1-17 所示。



图 1-16　添加监视



图 1-17　监视窗口

在【添加监视】对话框中可以设置监视的类型。监视类型包括监视表达式（默认）、当监视值为真时中断、当监视值改变时中断。后两种可以设置监视断点，当所监视的表达式为真或者改变的情况下中断程序的执行，以利调试。

1.4　从宏开始学习 VBA

使用 Excel 2007 VBA 做应用开发时，一个比较实用的功能之一就是宏。通常当对一个应用做过分析后，可以发现有部分代码是可以通过录制宏来获取的。适用宏的情况很多，如所编写的应用经常涉及到单元格格式设置以及大量的 Excel 本身内置功能的调用时。例如自动筛

选、高级筛选、排序、数据透视表等。采用首先录制宏然后修改代码的方式可以大幅度减轻开发的工作量，缩短开发的周期。

1.4.1 了解宏

宏是存储了一系列命令的程序。当创建一个宏命令时，只是将一系列的键盘输入组合成一个简单的命令。这些命令的组合在以后可以被重复“回演”。使用宏命令可以减少复杂任务的步骤，可以显著地减少花在创建、设置格式、修改和打印工作表的时间。用户可以通过Excel 2007 内置的录制工具来创建宏命令，也可以在代码编辑器里面直接书写宏代码。

通常当需要反复地做一些动作或 Excel 2007 没有提供一个内置工具帮助完成某个任务时，则可以选择创建一个宏。宏命令能够将工作表的部分工作自动化，例如创建一个宏命令去自动修改工作簿中工作表的标签、帮助检查选中的工作表区域里的重复值、快速地将某种单元格格式应用到多个单元格区域等。

Excel 还拥有非常强大的图表功能。如果想要实现图表创建和格式设置自动化，使用宏命令一样行之有效。宏命令还可以帮助设置打印区域、页边距、页眉、页脚以及选择特殊的打印选项。

调用宏的方式是，在 Excel 2007 的菜单中依次选择【视图】|【宏】命令（如图 1-18 所示），然后打开下拉列表框，随后在弹出的宏菜单（如图 1-19 所示）中选择相应的操作。该弹出菜单下包含了 3 个菜单选项：查看宏、录制宏和使用相对引用。



图 1-18 宏

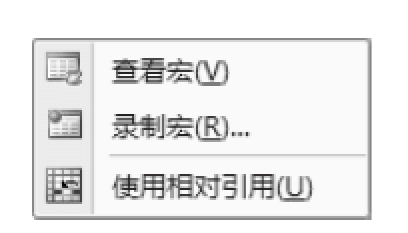


图 1-19 宏菜单

- 查看宏：选择该菜单选项后将打开宏管理窗口。在该窗口中可以完成运行、编辑、创建、删除、设置宏选项等宏管理工作。该窗口界面如图 1-20 所示。

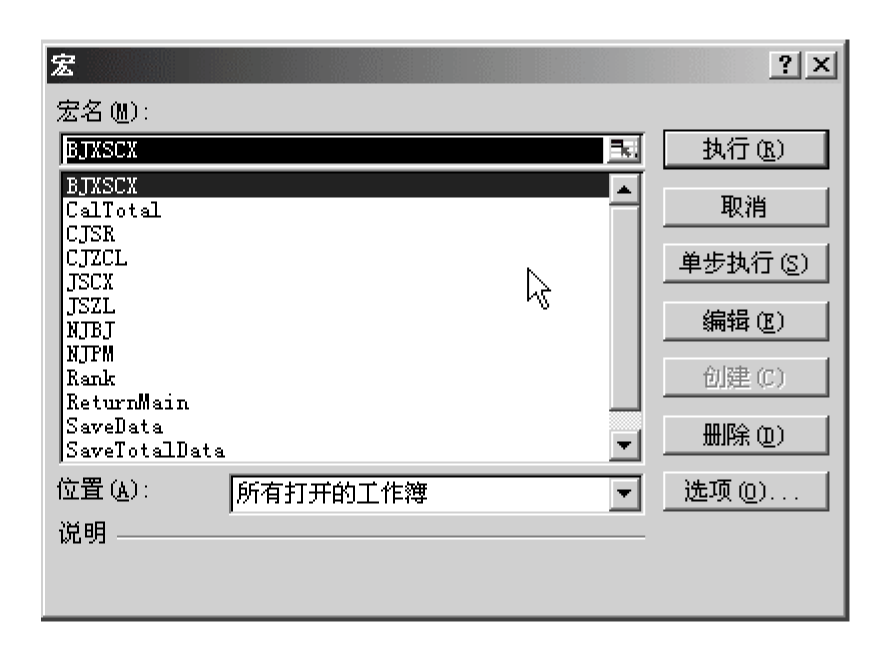


图 1-20 宏管理窗口

- 录制宏：选择该菜单选项后，首先弹出一个宏设置窗口（如图 1-21 所示），在该窗口中可以设置录制宏的名称以及快捷键。单击【确定】按钮后即开始录制宏。此时在 Excel 2007 的状态栏上出现【停止录制宏】按钮（如图 1-22 所示）。当录制完成

后，单击该按钮即可终止录制宏。



图 1-21　宏设置窗口

就绪

图 1-22　停止录制

1.4.2　录制宏实例

在创建一个宏命令之前，需要考虑究竟需要做什么。宏命令是一大堆键盘输入的集合，事先计划好所有动作非常重要。如果在录制宏之前，没有很好地计划，将会录制很多不必要的步骤。这些冗余的步骤将会影响宏的运行速度，也会给开发者编辑修改宏造成一定的不便。

笔者的建议是：在建立宏时，首先手动将宏命令需要做的事务演练一遍；同时，记录下每一步实际发生的动作，越精简越好；最后开始录制该宏并按照自己记录的步骤操作，减少宏代码冗余与操作错误带来的麻烦。

下面是一个录制宏的实例，该实例将学生表的 A 列（学生名列）的不重复项目筛选出来，然后填充到该表的 C 列。被记录下来的操作步骤如下：

（1）在菜单中选择【工具】|【宏】|【录制新宏】命令。

（2）选择“学生”表的 A 列。

（3）在菜单中选择【数据】|【筛选】|【高级筛选】命令，单击该按钮后，将打开【高级筛选】对话框，具体的设置如图 1-23 所示。



图 1-23　【高级筛选】对话框

该宏的代码如下：

```
Sub Macro1()
'Macro1 Macro
'宏由 Alex 录制，时间: 2007-9-8
    Columns("A:A").Select                                   '选择 A 列
    Range("A1:A22").AdvancedFilter Action:=xlFilterCopy, CopyToRange:=Range( _
        "C1"), Unique:=True                                 '筛选 A 列不重复记录到 C 列
'Action 参数：指定是否就地复制或筛选列表
'CopyToRange 参数：如果 Action 为 xlFilterCopy，则为复制行的目标区域；否则，忽略该参数
'Unique 参数：如果为 True，则只筛选唯一记录。如果为 False，则筛选符合条件的所有记录。默认值
为 False
End Sub
```

代码说明：

该代码中用到的 AdvancedFilter 方法可以基于条件区域从列表中筛选或复制数据。如果初始选定区域为单个单元格，则使用单元格的当前区域。该方法的各个参数的意义请参见本章的知识点。

1.4.3 分析与编辑宏代码

大多数情况下通过录制宏得到的代码通常是不适合被直接应用的，其中存在很多冗余的代码。这些冗余的代码通常会增加程序的运行时间。为了提高程序的运行速度与效率，降低维护难度，对录制宏得到的代码加以修改后使用很有实际意义。

在上述录制的宏代码中，选择 A 列的语句就是多余的。这是因为后续的高级筛选代码中指定了筛选的范围 Range("A1:A22")，因此无须再另行指定选择范围。另外其中的注释也没有必要，因而这段代码可以修改为如下代码：

```
Sub Macro1()
    Range("A1:A22").AdvancedFilter Action:=xlFilterCopy, CopyToRange:=Range( _
        "C1"), Unique:=True                                  '筛选 A 列不重复记录到 C 列
End Sub
```

除了代码中多余的注释与区域选择代码可以删除外，方法中有默认值的参数选项也可以去除。另外，很多通过录制宏获取的代码实际上可以将几步合并到一步中。这样的情况既可以精简代码、增强可阅读性，还可以提高程序的运行速度。例如录制复制/粘贴操作时，录制的代码类似以下代码：

```
Range("A1").Select                                  '选择 A1 单元格
Selection.Copy                                      '复制 A1 单元格
Range("F1").Select                                  '选择 F1 单元格
ActiveSheet.Paste                                   '粘贴 A1 单元格数据格式到 F1 单元格
```

这 4 行代码实际上可以使用以下一行代码来完成：

```
Range("A1").Copy Destination:=Range("F1")           '粘贴 A1 单元格数据格式到 F1 单元格
```

当对同一个对象执行多个操作时，可以使用 With…End With 结构，例如以下代码：

```
Range("A14:G14").Select                             '选择 A14:G14 单元格区域
Selection.Font.Bold=True                            '设置字体加粗
Selection.Font.Size=12                              '定义字体大小
Selection.Font.ColorIndex=5                         '定义文字颜色
Selection.Font.Underline=xlUnderlineStyleDoubleAccounting '定义下划线样式
```

以上代码可以换成：

```
With Range("A14:G14").Font
    .Bold=True
    .Size=12
    .ColorIndex=5
    . Underline=xlUnderlineStyleDoubleAccounting
End With
```

1.4.4　运行宏

要运行宏，可以使用在录制宏时设置的快捷键；也可以打开宏管理窗口（如图 1-20 所示），从中选择需要运行的宏，然后选择【运行】；还可以在 VBE 环境下打开该宏过程，然后单击 VBA 编辑器工具栏（如图 1-6 所示）中的【运行程序】按钮。在该环境中，可以逐行执行宏过程的代码。当读者无法读懂宏过程代码的意义时，可以采用逐行执行方式，检查每一步宏过程的运行情况。

第 2 章　VBA 程序设计基础

本章也是基础章节，重点讲解 VBA 程序设计的基础知识。本章依次讲解 VBA 各部分的基础，包括数据类型、常量、变量、过程与函数、运算符、结构语句、常用函数和数组。如果读者对这些内容已经有了较深的认识，可以跳过本章阅读后续实例章节。

2.1　数 据 类 型

数据类型决定了数据在计算机中的存储方式以及程序对该种类型数据所能做的操作。数据类型不同则其在内存中的存储结构不同，占用空间也不同。VBA 提供的基本数据类型主要有数值型、字节型、字符串型、逻辑型、日期型、无符号型、变体型、对象型等类型。如果没有定义变量的数据类型，VBA 将自动设置该变量的数据类型为变体型（Variant）。Variant 型有能力解决数据本身的操作类型并且使用该类型。表 2-1 中列出了 VBA 支持的所有数据类型。

表 2-1　VBA 基本数据类型

数据类型（名称）	大小（字节）	描　　述
Boolean	2	逻辑值 True 或 False
Byte	1	0 到 255 的整数
Integer	2	-32768~32767 的整数
Long	4	-2147483648~2147483647 的整数
Single	4	单精度浮点数值 负数：-3.402823E38~-1.401298E-45 正数：1.401298E-45~3.402823E38
Double	8	双精度浮点数值 负数：-1.79769313486231E308~-4.94065645841247E-324 正数：4.94065645841247E-324~1.79769313486231E308
Currency	8	（放大的整数（作者：整数除以 10000 得到的数值，参见 VBA 帮助））使用在定点计算中： -922337203685477.5808~922337203685477.5807
Decimal	14	+/-79228162514264337593543950335 没有小数点； +/-7.9228162514264337593543950335 小数点后有 28 位数字； 最小的非 0 数字是 +/-0.0000000000000000000000000001
Date	8	从 100 年 1 月 1 日到 9999 年 12 月 31 日的日期

续表

数据类型（名称）	大小（字节）	描　　述
String（变长字符串）	10 字节+字符串长度	变长字符串最多可包含大约 20 亿（2^31）个字符
Object	4	对象变量用来引用 Excel 中的任何对象
Variant（带数字）	16	最高范围到 Double 类型的任何数值
Variant（带字母）	22 字节+字符串长度	和变长字符串的范围一样
用户定义类型（使用 Type）	成员所需的数值	每个成员的范围和其数据类型的范围一致

VBA 除了基本数据类型之外，还可以由用户定义自己的数据类型。不同的数据类型占据电脑内存的空间是不一样的。为了保存内存并确保程序运行更快，应该选择占用字节最少的、同时又能处理当前数据的数据类型。

2.1.1　数值型

VBA 的数值型类型包括整型（Integer）、长整型（Long）、单精度浮点型（Single）、Double（双精度浮点型）和货币型（Currency）。

1．整型（Integer）

用于表示-32768~32767 之间的整数。存储该类型数据占用 2（16 位）个字节空间，其运算速度较快。定义该类型变量时，可以使用以下两种方式，其中第二种方式使用了类型声明符“%”。

```
Dim rowsCount as Integer
Dim rowsCount%
```

2．长整型（Long）

用于表示-2147483648~2147483647 之间的整数。存储该类型数据需要占用 4 个字节。长整型（Long）的类型声明符是“&”。

3．单精度浮点型（Single）

用于表示带小数的实型数，有效位为 7 位。存储该数据类型占用 4 个字节。单精度浮点型（Single）的类型声明符是“!”。一般以科学计数法表示，其中指数部分以“E”或“e”表示。其科学计数法表现形式举例如下：

4.55E+14 表示 4.55×10^{14}，3.69e-7 表示 3.69×10^{-7}。

4．双精度浮点型（Double）

同单精度浮点型，也表示带小数的实型数，有效位为 15 位。存储该数据类型需要占用 8 个字节。双精度浮点型（Double）的类型声明符是“#”。其科学计数法用“D”或“d”表示指数部分。

5．货币型（Currency）

用于货币计算。该数据类型支持 19 位有效数字，其中小数点右面占用 4 位有效数。货币

型（Currency）的类型声明符是“@”。

2.1.2　字节型（Byte）

字节数据类型用 1 字节存储 0~255 间的无符号单精度整型数值。如果变量用于二进制运算，应该声明为 Byte 数据类型数组。

2.1.3　字符串型（String）

字符串类型用于存储字符序列的数据类型，其定界符为一对英文双引号。在 VBA 中字符串分为两种：变长和定长字符串。字符串类型的类型声明符是“$”。

定长字符串的定义方式如下：

```
Dim rowsCount As String * 200
```

2.1.4　逻辑型（Boolean）

当数据只需要记录两种相反的状态信息时，该数据类型的数据占用 2 个字节。如表示“True/False”、“Yes/No”等，应该选择使用逻辑型（Boolean）。其默认值为 False。

2.1.5　日期型（Date）

日期型用于记录时间和日期数据。在 VBA 中日期数据类型的定界符为“#”。下面是一些实例：

- #月/日/年#：例如#10/23/2007#表示 2007 年 10 月 23 号。
- #年-月-日#：例如#2004-1-29#表示 2004 年 1 月 29 号。
- #年，月，日#：例如#1995,6,29#表示 1995 年 6 月 29 号。
- #英文月份缩写日，年#：例如#July 5,1998#表示 1998 年 7 月 5 号。
- #时:分:秒 AM/PM#：例如#10:30:54 AM#表示上午 10 点 30 分 54 秒。
- #月/日/年时:分:秒 AM/PM#：该格式是时间与日期两部分的结合。

2.1.6　无符号型（Decimal）

无符号类型（Decimal），该种数据类型存储 96 位（12 个字节）无符号的整数形式，并除以一个 10 的幂数，称为变比因子。这个变比因子决定了小数点右面的数字位数，其范围为 0~28。变比因子为 0（没有小数位）的情形下，最大的可能值为+/−79228162514264337593543950335。而在有 28 个小数位的情况下，最大值为+/−7.9228162514264337593543950335，而最小的非零值为+/−0.0000000000000000000000000001。

Decimal 数据类型只能在 Variant 中使用，也就是说，不能声明一变量为 Decimal 类型。不过可以用 Cdec 函数创建一个子类型为 Decimal 的 Variant。

2.1.7 变体型（Variant）

所有没有被显示声明其数据类型的变量，VBA 都将其作为变体型（Variant）处理。系统将会自动判断保存数据的类型。另外，该数据类型还包含了 Empty、Null 和 Error 3 个特殊值。

- Empty 值：当 Variant 数据类型没有被初始化前，其值为 Empty。该值不等价于 0、零长度字符串或 Null 值。
- Null 值：指定变量不含有效值，通常表示未知数据或丢失数据。
- Error 值：用于指示在过程中出现错误时的特殊值。与其他类型错误不同，这些错误并非发生正常的应用程序级的错误处理，因此，可以根据 Error 值对类型错误进行取舍。

2.1.8 对象型（Object）

对象类型（Object）用来表示图形、OLE 对象或其他对象，用 4 个字节存储。

2.1.9 用户自定义型

用户自定义型用于建立满足用户特殊需求的自定义数据类型，该数据类型由多个基本数据类型的数据组成。定义自定义数据类型采用 Type 语句。一般的格式如下：

```
Type 自定义数据类型名称
    元素 1 名称 As 数据类型
    元素 2 名称 As 数据类型
    …
    元素 N 名称 As 数据类型
End Type
```

2.2 常　　量

在程序执行前值已经确定，且执行过程中不能改变的量称为常量。VBA 中的常量包括直接常量、符号常量和系统常量。

2.2.1 直接常量

程序代码中直接书写的量就是直接常量。直接常量包括数值常量、字符串常量、日期/时间常量和布尔常量。

1. 数值常量

数值常量是直接以数字体现的数值型数据。在 VBA 中可以使用类型标识符表示常量的数据类型（各种数值型数据的类型标识符参见 2.1.1 节具体说明）。例如：8.75#表示的是一个双精度数值。

2. 字符串常量

字符串常量可以包括数字、英文字母、特殊符号和汉字等可打印字符。定义时必须用双引号加以标示。例如：

```
"这是一个字符串常量实例"
```

如果字符串常量中包括了双引号，此时需要在有双引号的地方输入两次双引号，例如：

```
"货币型数据类型的类型声明符是："@"" "
```

其中 1、6 双引号用于界定字符串常量，2、5 双引号用于界定内部双引号，3、4 双引号是字符串常量中的双引号。

3. 日期/时间常量

日期/时间常量是直接体现的日期型数据，书写的格式参见日期型数据的书写方法。

4. 布尔常量

布尔常量是直接体现的布尔型数据，书写的格式参见布尔型数据的书写方法。

2.2.2 符号常量

如果在代码中需要多次重复使用某一个常量时，应当命名该常量。这样做一方面可以增强程序的可读性，另一方面如果需要修改常量值，可以快速修改，亦可降低出错率。符号常量必须在程序开始运行前被定义，并且在程序运行中，不能修改和重新赋值。

符号常量的定义方式如下：

```
Const 符号常量名称=符号常量表达式
```

Const 是定义符号常量的关键字。定义了符号常量后，在之后的程序代码中就可以使用符号常量名称来引用符号常量表达式的值。通常定义一个有实际意义、便于记忆的常量名，以减轻工作量。如果需要建立全局性的符号常量，应在模块声明部分将常量声明为 Public 型。例如：

```
Public Const PI=3.14159265358979
```

定义符号常量时，可以使用直接常量，也可以使用计算结果为数字或字符串的表达式，还可以使用前面定义过的符号常量。例如：

```
Public Const PI2=PI*PI
```

2.2.3 系统常量

Excel 2007 VBA 提供了一系列用于各种用途的符号常量，这些常量统称为系统常量。这

些常量在应用程序对象的方法和属性中使用。这些常量通常都以 xl、vb 开头。在对象浏览器中可以通过输入 xl、vb 获得以这些开头的系统常量的列表，或者输入具体的常量名称查询该常量对应的值。如图 2-1 所示是在对象浏览器中输入 vb 后查看 vbMsgBoxResult 的常量成员时的效果图。



图 2-1　查看系统常量

2.3　变　　量

变量是一个用来引用一条数据的量。通常在程序执行过程中，用变量存储临时数据，其内容随程序的执行而变化。变量可以被看作为存放未知值的内存单元。

2.3.1　变量命名

引用变量时是通过变量名称实现的。变量名称里可以包含字母、数字和一些标点符号，但不包含# $ % & @ !这 6 个符号。命名时需要遵循以下规则：

- 必须以字母开头，后跟字母、数字或下划线，在中文 Excel 2007 VBA 中还允许使用汉字。
- 不能有空格或使用类型声明符等特殊符号。
- 长度不能超过 255 个字符。控件、窗体、类或模块的名字不能超过 40 个字符。
- 不能与系统关键字同名。

VBA 的变量名不区分大小写。在命名时通常采取统一的命名规则。通常做法是：用变量类型的缩写小写字母做前缀，后跟首字母大写的有意义的名称。例如：strName，表示 Name（姓名）的字符串变量；intAge，表示 Age（年龄）的整型变量。

注意： 关键字指在 VBA 中有特殊意义的单词或单词的缩写，如 Public、Sub 等。

2.3.2 变量声明

在使用变量之前，一般先对该变量加以声明，告知 VBA 该变量的数据类型其占用的存储空间。声明变量的方法如下：

```
Dim 变量名称 [As 数据类型]
```

Dim 和 As 是声明变量的关键字。中括号所标识的是可以省略的部分，即变量的数据类型可以不指定。在 VBA 中，声明变量包括隐式声明和显式声明两种方式。以下是这两种方式的详细介绍：

1. 隐式声明

在使用一个变量之前没有对该变量进行声明，这种变量的声明方式即为隐式声明。此时，VBA 会自动创建该变量，并将其类型设置为 Variant 类型，此时其值为 Empty。当其被赋值后，该值的数据类型将替代该变量的数据类型。

隐式声明变量并不是良好的编程习惯。程序中大量使用未被声明的变量，经常造成不必要的错误，加大程序维护与调试难度，也不便于阅读。例如变量的拼写错误，这些错误通常不能被编译系统检查出来。

2. 显式声明

变量使用 Dim 关键字加以声明，即为显式声明。为了避免隐式声明带来的麻烦，可以设置为当遇到未声明变量时 VBE 提示错误警告。方法参见 1.2.1 节关于 VBE 环境的设置的具体说明。

2.3.3 变量的作用范围

变量能够发生作用的范围即变量的作用范围。通常按照其作用范围可以把变量划分为局部变量、模块变量、全局变量和静态变量。

1. 局部变量

在过程的内部被声明的变量即为局部变量。该变量只能在其被声明的过程中有效。通常把过程中临时使用的变量声明为局部变量。这样不同过程中类似的变量就可以采用相同的变量名称，并且这些变量相互之间没有任何相互关联的关系，互不影响。

2. 模块变量

在模块的顶部声明的变量即为模块变量。该变量在该模块的所有过程中都是有效的，但在其他模块中不能被引用。

> 注意：当模块变量与该模块中某个过程的局部变量同名时，该过程中出现的变量名称引用的是局部变量，而非模块变量。

3. 全局变量

使用 Public 关键字在模块中声明的变量即为全局变量。该变量在所有模块中有效，程序中所有过程都可以访问该变量。

当变量在程序中的多个模块中共享时，需要将其声明为全局变量。如果变量需要在同一模块的多个过程中共享时，应当声明为模块变量。尽量多地使用局部变量，在必要的情况下使用全局变量和局部变量，以减少程序的开销以及出错几率。

注意：不能在过程中声明全局变量，只能在模块的声明部分声明全局变量。

4. 静态变量

对于一般的局部变量，当包含该变量的过程执行完毕后，该变量的值将不会继续存在，其占用的内存也将被系统释放。下一次执行该过程时，系统将会给该局部变量重新分配内存单元并进行初始化。

但是有时候，需要过程中的局部变量具有记忆能力。当过程结束后，我们希望该变量的值能够继续保存下去。例如有某个按钮，现在希望该按钮被按奇数次时完成与被按偶数次时不同的任务，这个时候就需要使用静态变量。

当局部变量被定义为静态变量后，就可以在退出过程时保存该变量的值，以供再次调用该数据。静态变量的定义格式如下：

```
Static intClickCount As Integer
```

下面是一个静态变量使用的实例。该实例对双击表 Sheet1 的 A1 单元格的次数做统计。当双击偶数次时，设置 A1 单元格无填充；双击奇数次时设置 A1 单元格为红色填充色。

Sheet1 双击事件代码如下：

```
Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)
If Target.Row = 1 And Target.Column = 1 Then                '检测是否双击 A1 单元格
    ClickCount                                              '双击 A1 单元格时执行 ClickCount 过程
End If
End Sub
```

ClickCount 过程代码如下：

```
Sub ClickCount()
Static intClickCount As Integer                             '定义静态局部变量保存双击次数
intClickCount = intClickCount + 1                           '累计双击次数
If intClickCount Mod 2 Then
    Sheet1.Range("A1").Interior.Color = RGB(255, 0, 0)      '如果双击奇数次设置红色填充色
Else
    Sheet1.Range("A1").Interior.Pattern = xlNone            '如果双击偶数次清除填充格式
End If
MsgBox "您已双击 A1 单元格" & intClickCount & "次！"          '给予双击次数提示信息
End Sub
```

2.4 认识过程与函数

过程是模块中最小的单元，通常用于完成某个相对独立的功能。使用过程可以增强程序的结构性，使程序的流程更加清晰，也利于程序的维护和升级。VBA 中常用的过程包括 Sub 过程和 Function 函数，下面分别介绍这两个过程类型。

2.4.1 Sub 过程

Sub 过程是没有任何返回值的一段程序，Sub 过程又被称为子过程。通常 Sub 过程用于完成一个独立的、明确的任务。在 VBA 中响应事件的代码块都是 Sub 过程。

Sub 过程定义格式如下：

```
[Private|Public] [Static] 过程名称([ByVal|ByRef 参数名称 As 参数类型])
     局部变量声名
     过程代码
End Sub
```

1．Private|Public 关键字

将过程声明为公共过程或者私有过程。当被声明为 Private 时，只有在包含该过程的模块中其他过程可以访问或调用该过程；当被声明为 Public 时，所有模块的所有过程都可以调用这个过程。

如果没有使用该关键字，那么系统将默认使用 Public 设定该过程。

2．Static 关键字

当将过程声明为 Static 后，该过程中声明的所有局部变量都成为静态变量，所有变量都具有了记忆能力。该关键字的限定作用不能对 Sub 过程外的变量起作用，包括那些在该过程中使用了但并没有在该过程中声明的变量。

3．ByVal|ByRef 关键字

该类关键字限定参数的传递方式。ByRef 为默认参数传递方式，该方式传递的是参数的引用，过程用变量的内存地址访问实际变量的值，因而在过程中可以直接修改实际变量的值；ByVal 指定参数按值传递，传递的是变量的一个内存副本，过程中对该副本的修改不会影响到实际变量。

> **注意:** 所有的可执行代码都必须处于某个过程中，否则无法执行。不能在过程或函数内部定义另外的过程。

在 VBE 开发环境下创建过程可以采用两种方法。

❑ 菜单添加过程

首先在【工程资源管理器】窗口中双击对应的模块，打开需要添加过程的模块代码编辑

器窗口。然后选择【插入】|【过程】命令，打开【添加过程】对话框（如图 2-2 所示）。在【名称】文本框中输入过程名称后单击确定按钮，系统将会自动生成该子过程的结构。

如果生成的过程需要接收参数，则在【代码编辑器】窗口中过程名后的括号中加入相应的参数传递方式、参数名、数据类型。

添加过程
名称(N): MySub
确定
取消
类型
子程序(S)
函数(F)
属性(P)
范围
公共的(B)
私有的(V)
把所有局部变量声明为静态变量(A)

图 2-2　【添加过程】对话框

❑ 代码添加过程

代码添加过程也十分简便。添加过程时，需要做的事情是在代码区新的一行按 Sub 过程定义格式依次输入即可。输入完过程的反括号后按 Enter 键，VBE 将自动添加过程结束语句 End Sub。

2.4.2　Function 过程

Function 过程是函数过程，简称函数。函数用于建立用户自定义函数，自定义函数的调用和 Excel 2007 的内部函数调用方法一致。其与 Sub 过程类似的是，Function 过程也是一个独立的过程。函数可以带有参数列表，执行一系列语句，改变参数的值，但是 Function 过程可以返回一个值作为其被调用的结果。

定义函数过程的语法格式如下：

```
[Public|Private|Friend] [Static] 函数名称([ByVal|ByRef 参数名 As 参数数据类型]) As 返回值数据类型
    局部变量声明
    函数内部代码
    函数结果赋值语句
End Function
```

函数结果赋值语句用于将最后处理的结果赋值给函数，这个结果将作为函数的返回值。Function 过程的一般调用方式如下：

```
函数名称=表达式
```

在 VBE 环境下，创建 Function 过程和 Sub 过程的创建一致，不同的是在“添加过程”对话框中，选择过程类型时应该选中【函数】单选按钮。

2.5　表达式与运算符

表达式是由运算符和操作数共同组成的合法算式。操作数可以是常数、变量、函数或表达式。运算符是连接操作数的运算符号。VBA 提供了 5 种类型的运算符号：算术运算符、比较运算符、逻辑运算符、连接运算符和特殊运算符。

2.5.1 算术运算符

算术表达式包括基本的算术运算符号。这些运算包括加减乘除、指数运算、负数运算、整除与取模运算。这些运算间存在运算的优先级，表 2-2 是各种运算符号及其运算优先级。当需要改变运算次序时，可以使用圆括号改变运算的优先次序。

表 2-2 算术运算符及其优先级

算术运算符	优 先 级	描 述	实 例
^	1	指数运算	4^2
-	2	负数运算	−100，−200
*	3	乘法运算	4*6=24
/	3	除法运算	9/2=4.5
\	4	整除运算	101\4=25
mod	5	取模运算	7 mod 3=1,10 mod 4=2
+	6	加法运算	5+3=8
−	6	减法运算	10−3=7

2.5.2 比较运算符

比较运算符是用来比较两个数或表达式的运算符，它的主要作用是确定比较双方的关系。运算的结果可分为 True、False 和 Null，只要运算的双方中有任何一方是 Null，结果还是 Null。该运算符又叫关系运算符。表 2-3 列出了各个比较运算符及其功能。因为比较运算的运算优先级别一致，表中没有列出优先级。

表 2-3 比较运算符及其功能

比较运算符	作 用	实 例
=	等于比较	x=y
<>	不等于比较	x<>y
<	小于比较	x<y
>	大于比较	x>y
<=	小于等于比较	x<=y
>=	大于等于比较	x>=y

2.5.3 逻辑运算符

逻辑运算是指表达式间的逻辑关系运算。逻辑运算符通常用来表示比较复杂的关系，其最终运算结果只可能是 True 或 False。逻辑运算符具有不一致的运算优先级。表 2-4 列出了各个逻辑运算符以及其运算优先级。

表 2-4 逻辑运算符及其优先级

逻辑运算符	优 先 级	描 述	实 例
NOT	1	取反运算	NOT x
AND	2	逻辑与运算	x AND y
OR	3	逻辑或运算	x OR y
XOR	4	逻辑异或运算	x XOR y
EQV	5	逻辑相等运算	x EQV y
IMP	6	逻辑蕴涵运算	x IMP y

表 2-5 体现了逻辑运算的结果。该表假定 x 和 y 为任意两个操作数，用 T 表示逻辑真，用 F 表示逻辑假。

表 2-5 逻辑运算符真值表

操作数 x	操作数 y	NOT x	NOT y	x AND y	x OR y	x XOR y	x EQV y	x IMP y
F	F	T	T	F	F	F	T	T
F	T	T	F	F	T	T	F	T
T	F	F	T	F	T	T	F	F
T	T	F	F	T	T	F	T	T

2.5.4 连接运算符

连接运算符就是将两个表达式连接在一起的运算符号。在 VBA 中用于连接的运算符包括“+”和“&”。“+”主要用于连接两个都是字符串的情况；“&”运算符将两个运算数强制转换为字符串后连接。表 2-6 列出了两个连接运算符的区别。

表 2-6 &与+连接运算差别表

操作数 x	操作数 y	&连接的结果	+连接的结果
"123"	"3"	"1233"	"1233"
123	3	"1233"	126
"123"	3	"1233"	126
"123a"	3	"123a3"	报错
"hello"	"World"	"hello World"	"hello World"

2.5.5 特殊运算符

VBA 提供了两种特殊运算符：Is 和 Like，它们应属于比较运算符。以下分别介绍这两个运算符的使用方法。

1. Is 运算符

这个运算符号用于比较两个对象变量的引用变量是否一致，返回结果为 True 或 False。使



用该运算符的语法格式如下：

```
结果=对象 1 Is 对象 2
```

如果对象变量“对象 1”和“对象 2”两者引用相同的对象，则结果为 True；否则结果为 False。以下是该运算符使用的一个实例：

```
Dim MyObject,YourObject,ThisObject,OtherObject,ThatObject,MyCheck        '定义变量
    …                                                                '给变量赋值
Set YourObject=MyObject                                              '指定对象引用
Set ThisObject=MyObject
Set ThatObject=OtherObject                                           '假设 MyObject<>OtherObject
MyCheck=YourObject Is ThisObject                                     '返回 True
MyCheck=ThatObject Is ThisObject                                     '返回 False
```

2．Like 运算符

该运算符把一个字符串表达式与一个给定模式（SQL 表达式中的样式）进行匹配。匹配成功返回结果 True，否则返回结果 False。该运算符主要用于数据库查询过程中，其使用的语法格式如下：

```
结果=字符串变量 Like 模式
```

如果字符串变量与模式表达式匹配，则结果为 True，否则结果为 False。但是如果字符串变量或模式中有一个为 Null 时，则结果为 Null。Like 运算符内建的模式匹配功能提供了多种方式来进行字符串比较。有的模式匹配功能就可以使用通配符、字符列表或字符区间的任何组合来匹配字符串。表 2-7 列出了常用的匹配符号。

表 2-7　Like 运算符中模式设置表

模式中的匹配字符	字符串变量中相应的匹配内容
?	任何单一字符
*	零个或多个字符
#	任何一个数字（0~9）
[字符序列]	字符序列中的任何单一字符
[!字符序列]	不在字符序列中的任何单一字符

在中括号（[ ]）中，可以用由一个或多个字符组成的集合与字符串变量中的任一字符进行匹配，这个集合几乎可以包括所有字符和数字。

通过在范围的上、下限之间用连字符（-），字符列表可以指定字符的范围。例如，如果字符串变量中相应字符的位置包括 A~Z 之间的任意大写字母，则[A-Z]得到一个匹配模式。不需要分界符的情况下，方括号内就可以包括多个范围。

2.6 结构语句

VBA 的过程大部分不会一行一行从开始执行到末尾。经常需要在执行过程中跳转到过程

中的其他语句，以便控制程序的流程。比如有些过程中需要通过检测某个变量，然后再决定下一步应该执行哪一部分语句。此时就需要使用 If 结构语句调整程序的运行结构，使程序的执行流程符合实际目的。

本节讲解的结构控制语句包括：赋值语句、输出语句、If…Then 语句、If…Then…Else 语句及其变体、Select Case 多分支结构语句、Do…Loop 循环语句、For…Next 语句、For Each…Next 语句和跳转语句。

2.6.1　赋值语句

赋值语句是将表达式的结果赋给变量的语句。赋值语句的格式如下：

```
[Let]  变量名=表达式
```

其中 Let 关键字可以省略。使用该方式赋值时需要注意以下几点：

- 此处的“=”并不是比较运算符，其意义并不是“等于”，只起赋值操作，其左边只能是变量，不能使用表达式。
- 变量的数据类型必须与右边表达式最终计算结果数据的数据类型兼容，否则不能完成赋值操作。例如：

```
Dim intCount as integer
intCount= "表达式"
```

此处第二句的赋值操作是错误的，因为该语句将字符串数据赋值给了整型数据。

2.6.2　输出语句

在 Excel VBA 中输出时可以将结果直接输出到 Excel 工作表中，也可以将结果使用 Debug.Print 输出到立即窗口中。通常输出到立即窗口中可以便于调试，该语句在单步调试模式中也可以查询当前内存变量的数据值，以便于调试。输出到 Excel 工作表的方法后面章节有具体介绍。输出到立即窗口的语句格式如下：

```
Debug.Print [表达式] [分隔符]
```

Debug.Print 计算表达式的值，然后将其值输出到立即窗口中。有多个表达式时，使用分隔符分隔开多个表达式。以下是各种分隔符的形式：

- Spc(n)：在输出表达式的结果数据间插入 n 个空格。
- Tab(n)：在输出表达式的结果数据间插入 n 个制表符。
- 分号：将多个输出表达式的结果数据用分号连接后输出。
- 逗号：以 14 个字符为一个输出单元，每个表达式的结果数据输出到对应的输出单元内。

2.6.3　If…Then 语句

If…Then 语句用于有条件地执行一条或一段语句。If…Then 语句是典型的分支结构语句之

一。该语句有两种语法格式。

1．单行条件结构语句

单行条件结构语句的格式如下：

```
If 逻辑表达式 Then 语句
```

逻辑表达式可以是任何计算数值的表达式。VBA 将以 True 或 False 表示该逻辑表达式的计算结果，对于非零值都被看作 True，而为零的表达式被看作 False。

该语句的执行过程为：如果逻辑表达式为 True 时，将执行 Then 关键字后的语句；否则不执行 Then 后语句，而直接跳转到下一条语句。该过程如图 2-3 所示。

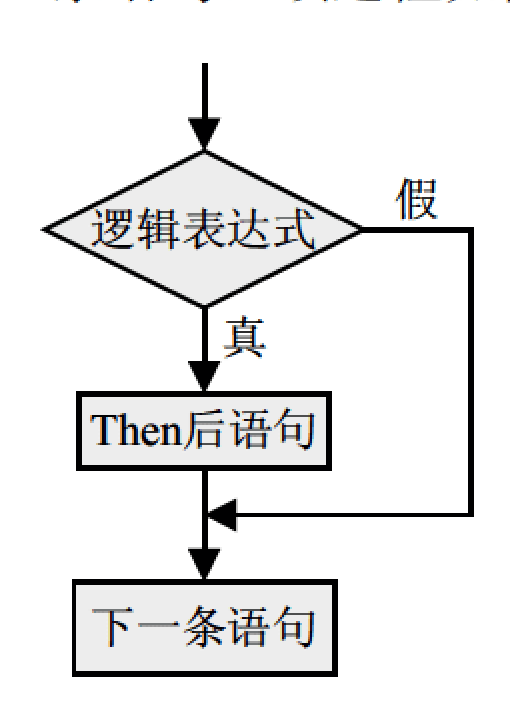


图 2-3　If…Then 语句流程图

例如：

```
If sales = 10000 AND salary <45000 Then SlsCom = Sales * 0.07
```

2．多行条件语句

当逻辑表达式为真时需要执行多行代码时，需要使用多行条件语句。其结构如下：

```
If 逻辑表达式 Then
    语句块
End If
```

单行条件结构语句和多行条件语句的区别是：单行条件结构语句最后不需要 End If 语句来终止整个条件结构，而多行条件语句需要使用 End If 标识条件结构的终结。

例如：

```
If price = 7 AND units >= 50 Then
    rebate = (price * units) * 0.1
    Range("A4").Value = "The rebate is: $" & rebate
End If
```

2.6.4　If…Then…Else 语句及其变体

在上面介绍的 If…Then 语句对于逻辑表达式为 False 时，没有提供可执行的语句代码块；当在逻辑表达式为 False 时要执行另外的代码，就需要使用 If…Then…Else 语句。

该语句也有两种格式：

1．单行条件结构语句

单行条件结构语句的书写格式如下：

```
If 逻辑表达式 Then 语句 1 Else 语句 2
```

该语句的执行过程为：当逻辑表达式为 True 时，执行语句 1；当逻辑表达式为 False 时，执行语句 2。该过程如图 2-4 所示。

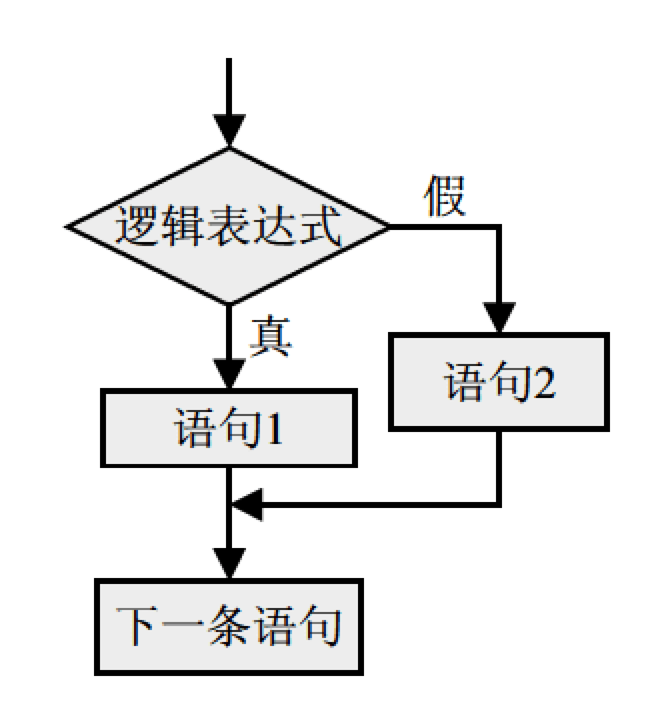


图 2-4 If…Then…Else 语句流程图

例如：

```
If Sales>5000 Then Bonus = Sales * 0.05 Else MsgBox "No Bonus"
```

2．多行条件结构语句

当要执行多个语句时，应该使用多行格式的 If…Then…Else 语句。该语句的格式如下：

```
If 逻辑表达式 Then
    语句序列 1
Else
    语句序列 2
End If
```

例如：

```
If ActiveSheet.Name = "Sheet1" Then
    ActiveSheet.Name = "My Sheet" MsgBox "This sheet has been renamed."
Else
    MsgBox "This sheet name is not default."
End If
```

该语句还有一种变体格式，其格式如下：

```
If 逻辑表达式 1 Then
    语句序列 1
ElseIf 逻辑表达式 2 Then
    语句序列 2
Else
    语句序列 N
End If
```

该变体格式可以为 If…Then…ElseIf 语句增加一个分支，因此该变体格式通常用于有两个以上条件分支时。

2.6.5 Select Case 多分支语句

当分支结构较为繁杂时，应该考虑使用 Select Case 多分支语句。虽然可以使用 If…Then…ElseIf 语句，但当所有的 ElseIf 语句都使用相同的逻辑表达式结构时，其结构将十分繁杂，且不易阅读，维护困难。

Select Case 多分支语句结构的语法格式如下：

```
Select Case 测试表达式
    Case 表达式列表 1
        语句序列 1
    Case 表达式列表 2
        语句序列 2
    Case 表达式列表 N
        语句序列 N
    Case Else
        没有表达式匹配测试表达式时执行的语句
End Select
```

该表达式在其开始处计算测试表达式，然后将表达式的值与结构中的每个 Case 值进行比较。如果相等，就执行与该 Case 相关联的语句块，执行完毕后跳转到 End Select 语句后执行其他语句。如果比较完所有 Case 语句都没有匹配的分支，就会执行 Case Else 分支后的语句代码。当不止一个 Case 语句与测试表达式相匹配时，则只执行第一个匹配的 Case 结构相关的语句序列。其执行过程图如图 2-5 所示。

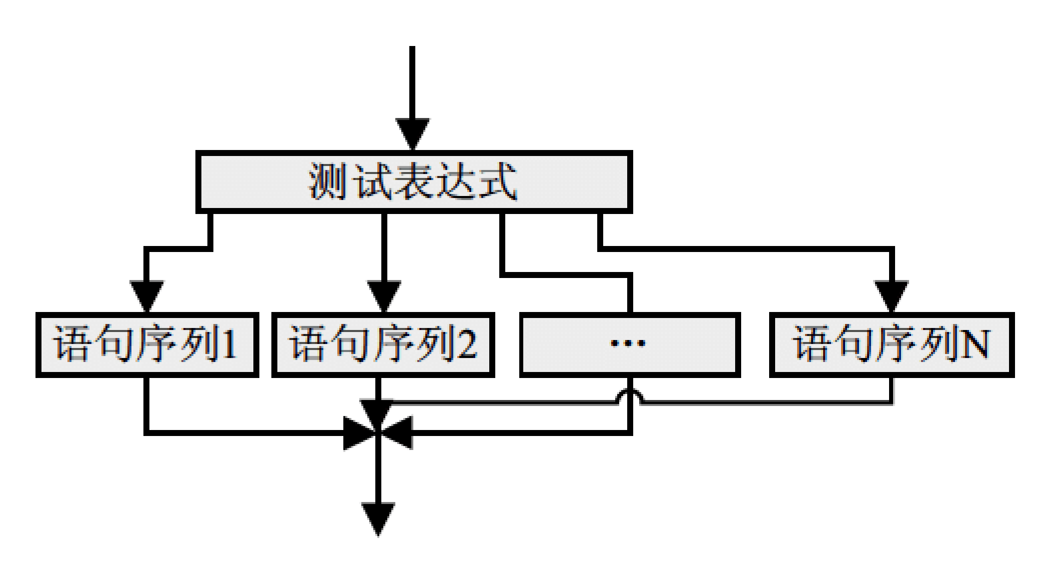


图 2-5 Select Case 语句流程图

其中测试表达式可以是数值型或字符型的表达式，通常是一个该数据类型的变量。表达式列表可以是一个或者几个值的列表，如果在一个列表中有多个值，就用逗号隔开。

表达式列表的书写具体有以下几种情况：

- ❑ 具体值形式：该表达式主要用于体现一系列具体的值。例如，Case “张”，“刘”，“李”。
- ❑ 取值范围形式：该形式用来表示一个数据范围。例如，Case 23 To 45。
- ❑ 使用 Is 运算符：例如 Case Is<200 表示所有小于 200 的值。

以上 3 种格式可以结合使用。以下是 Select Case 分支结构的一个实例：

```
Select Case myButton
    Case 6
        Workbooks.Add
    Case 7
        MsgBox "You can open a new book manually later."
    Case Else
        MsgBox "You pressed Cancel."
End Select
```

2.6.6　Do…Loop 语句

前面介绍的结构语句都只能让语句顺序执行一次。在实际处理过程中，经常需要用同种方法对各个数据进行重复处理。针对这种需要重复执行具有特定功能程序段的程序需要使用循环结构。

Do 循环语句有 4 种变体，即 Do While…Loop、Do…Loop While、Do Until…Loop 和 Do…Loop Until 语句。只要（或者直到）某个条件为真，就会重复一系列的语句。这些循环语句重复运行代码的次数并不一致，需要视情况而定，但每种循环都需要计算条件表达式的值。

1．Do While…Loop 语句

该语句的一般语法格式如下：

```
Do While 逻辑表达式
    语句序列
Loop
```

当该语句被执行时，首先判断逻辑表达式。如果为 False，则跳出所有语句，执行 Loop 语句后的其他语句；如果为 True，则执行循环体，执行到 Loop 后又跳转到 Do While 语句再次对逻辑表达式进行计算。在该结构中可以使用 Exit Do 语句结束循环，跳转到 Loop 后的语句继续执行。该语句的执行顺序如图 2-6 所示。

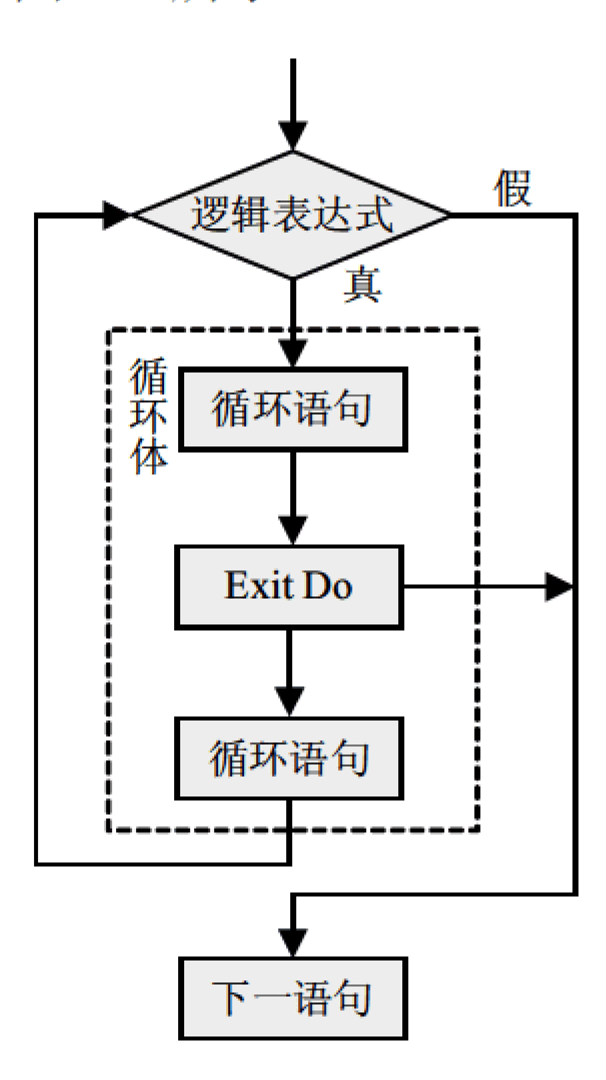


图 2-6　Do While…Loop 语句流程图

由于该循环结构在进入循环体前首先对逻辑表达式进行计算，如果逻辑表达式计算结果为 False，则跳出整个循环，所以该结构的循环体可能一次都不会被执行。

下面是该结构的一个实例：

```
Do While Now < stopme
    Application.DisplayStatusBar = True
    Application.StatusBar = Now
Loop
```

2．Do…Loop While 语句

Do…Loop While 语句和 Do While…Loop 唯一不同的地方是，该语句是首先执行一次循环体之后才开始检测逻辑表达式，因此该语句中循环体至少会被执行一次。其语法结构如下：

```
Do
    语句序列
Loop While 逻辑表达式
```

其执行流程图如图 2-7 所示。

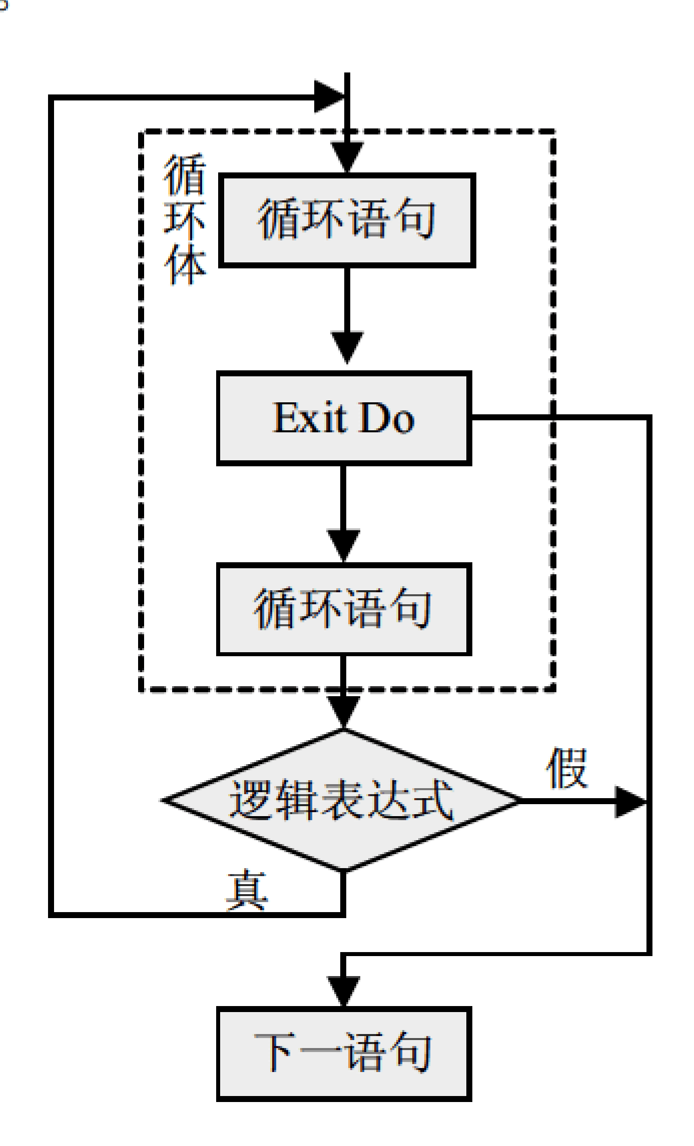


图 2-7　Do…Loop While 语句循环流程图

下面是该语句的一个实例。

```
Do
    secretCode = InputBox("Enter your secret code:")
    If secretCode = "sp1045" Then Exit Do
Loop While secretCode <> "sp1045"
```

3．Do Until…Loop 语句

该结构与 Do While…Loop 语句不同之处是：当 Until 后的逻辑表达式为假时该语句执行循环体；当逻辑表达式为真时，跳出循环。此语句将计算逻辑表达式置于语句开始，所以循环体可能一次都未被执行。

该语句的语法结构如下：

```
Do Until 逻辑表达式
    语句序列
Loop
```

其运行的流程图只需要将 Do While…Loop 语句的流程图（如图 2-6 所示）中对逻辑判断结果——真假互换位置即可。

下面是该语句的一个实例。

```
Do Until IsEmpty(ActiveCell)
    ActiveCell.Font.Bold = True
    ActiveCell.Offset(1, 0).Select
Loop
```

4．Do…Loop Until 语句

该语句的语法结构如下：

```
Do
    语句序列
Loop Until 逻辑表达式
```

该语句和 Do Until…Loop 语句类似，不同的是：当 Until 后的逻辑表达式为假时该语句执行循环体；当逻辑表达式为真时，跳出循环。此语句将计算逻辑表达式置于语句开始，所以循环体可能一次都未被执行。

下面是该语句的一个实例。

```
Do
    Worksheets(shcount).Select
    Set myRange = ActiveSheet.UsedRange

    If myRange.Address = "$A$1" And Range("A1").Value = "" Then
        Application.DisplayAlerts = False
        Worksheets(shcount).Delete
        Application.DisplayAlerts = True
    End If
    shcount = shcount -1
Loop Until shcount = 1
```

2.6.7　For…Next 语句

For…Next 语句以指定次数来重复执行循环体。与 Do 循环不同，For 循环使用一个计数器的变量。每次循环之后，计数器变量的值就会增加或减少一个固定值。在 For 循环中可以使用 Exit For 语句退出循环。For 循环的语法结构如下：

```
For 循环变量=初始值 To 终值 [Step 步长]
    循环体语句
Next [循环变量]
```

其中的步长可以为负值。当其为正值时，则初始值必须小于或等于终值，循环体才会被执行。当其为负值时，则初始值必须大于或等于终值，循环体才会被执行。该步长的默认值为 1。For…Next 循环结构的执行流程图如图 2-8 所示。

该语句的执行流程如下：

（1）将初始值赋予循环变量。

（2）判断循环变量是否超过终值，若为真，则退出循环，执行 Next 后的语句。这里的超过终值有两种情况：如果步长为负值时，超过就是循环变量的值小于终值；反之，超过就是循环变量的值大于终值。

（3）执行循环体语句。

（4）循环体执行完后，将循环变量累加步长。

（5）跳转到前面的第二步。

For…Next 语句的循环次数可以计算出来，计算方法如下（中括号表示对结果取整）：

循环次数=[(|终值-初值|)/步长]+1

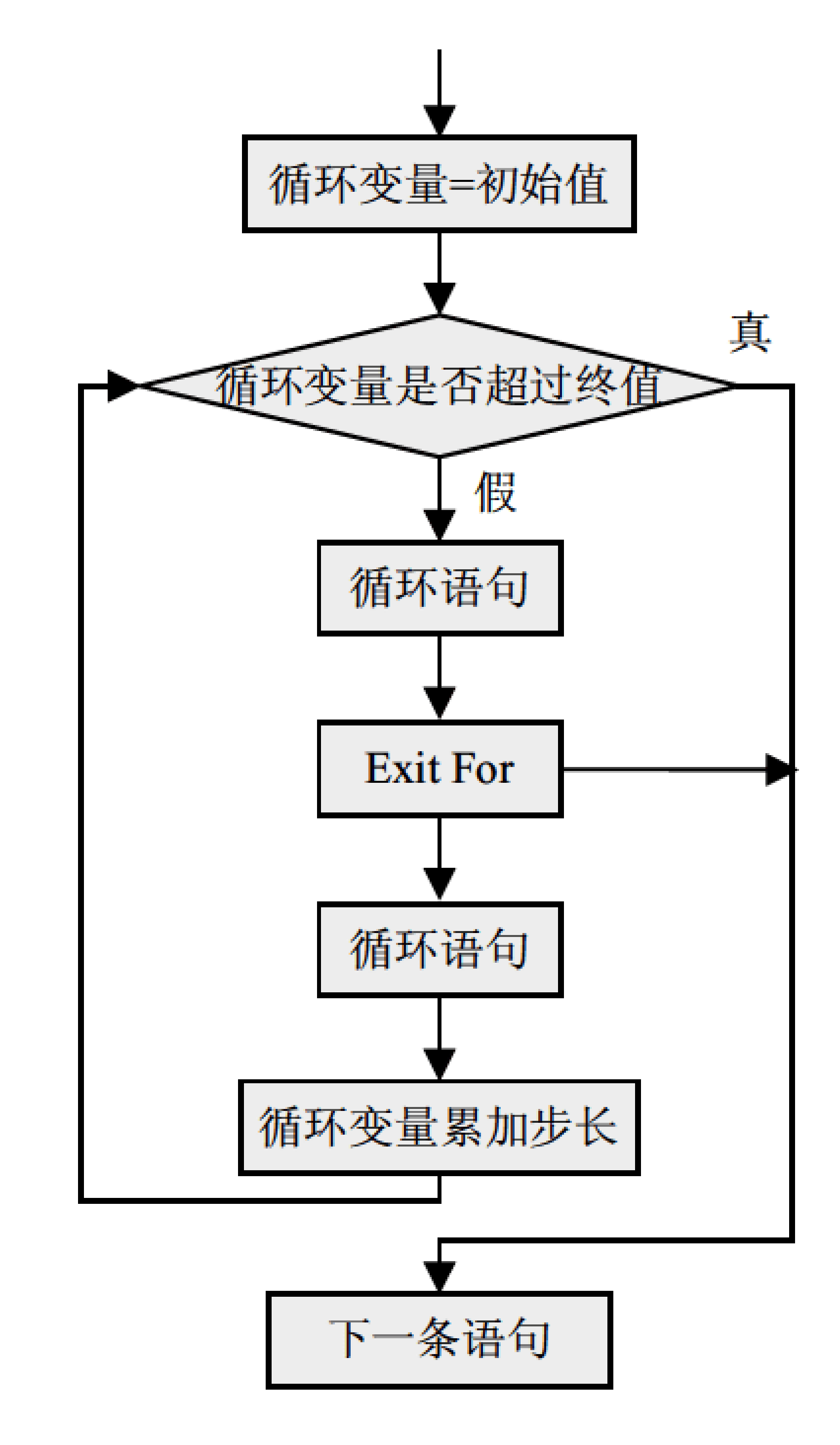


图 2-8　For…Next 语句流程图

注意：在事先不知道循环执行的次数时，应该采用 Do 循环结构；当知道循环次数时，应该采用 For…Next 循环结构。

以下是该语句的一个实例：

```
For r = 1 To totalR-1
    If ActiveCell = 0 Then
        Selection.EntireRow.Delete
        totalR = totalR-1
    Else
        ActiveCell.Offset(1, 0).Select
    End If
Next
```

2.6.8　For Each…Next 语句

For Each…Next 循环与 For…Next 循环类似，但该语句主要针对数组或对象集合中每一个元素重复执行一组代码的情况。当不知道一个集合中有多少个元素时，应该使用 For Each…Next 循环。该语句的语法结构如下：

```
For Each 对象元素变量 In 对象集合
    语句序列
Next [对象元素变量]
```

该语句使用时需要注意以下几点：

- 对象元素变量只能是 Variant 变量，或一般的 Object（对象）类型，或者是对象浏览器中列出的对象。
- 对于数组，对象元素变量之能是 Variant 变量。
- 不能使用用户自定义类型的数组，因为 Variant 数据类型不包含用户自定义数据类型。

注意：For 结构的语句，在 Next 后面都可以再跟一个变量。For…Next 后跟循环变量，For Each…Next 后跟对象元素变量，但是这些都可以省略。

以下是该语句的一个实例。

```
For Each mySheet In Worksheets
    ActiveWindow.SelectedSheets.Delete
Next mySheet
```

2.6.9　跳转语句

在 VBA 中可能需要代码的跳转执行。比如在循环结构的循环体运行过程中，有时候需要直接跳出循环体，继续执行循环结构后面代码。

当要跳转执行代码时，有以下几种办法。

1．使用 Exit 终止当前结构

使用 Exit 语句，将会终止当前结构，跳转到当前结构后面的执行代码。该语句包括 Exit Do、Exit For、Exit Function 和 Exit Sub 4 种表达方式。分别终止对应的结构。下面是一实例。

```
Sub ExitStatementDemo()
Dim I, MyNum
    Do                                  '建立无穷循环
        For I = 1 To 1000               '循环 1000 次
            MyNum = Int(Rnd * 1000)     '生成一随机数码
            Select Case MyNum           '检查随机数码
                Case 7: Exit For        '如果是 7，退出 For…Next 循环
                Case 29: Exit Do        '如果是 29，退出 Do…Loop 循环
                Case 54: Exit Sub       '如果是 54，退出子过程
            End Select
        Next I
    Loop
End Sub
```

2．GoTo 语句（On Error GoTo）

直接使用 GoTo 语句的情况不常见，为了维护程序结构性，应该尽量少使用 GoTo 语句。通常使用该方式的情况是针对错误处理。此时是在 On Error GoTo 语句后面跟跳转标记。下面是一个实例。

```
Sub GotoStatementDemo()
Dim Number, MyString
```

```
    Number = 1                          '设置变量初始值
    '判断 Number 的值以决定要完成哪一个程序区段（以“程序标签”来表示）
    If Number = 1 Then GoTo Line1 Else GoTo Line2

Line1:
    MyString = "Number equals 1"
    GoTo LastLine                       '完成最后一行
Line2:
    '下列的语句根本不会被完成。
    MyString = "Number equals 2"
LastLine:
    Debug.Print MyString                '将"Number equals 1"显示在立即窗口
End Sub
```

2.7 常见函数与语句

VBA 中除了各种控制结构语句以外，还提供了程序注释以及对话框形式的输入、输出语句。这些功能可以辅助开发者完善系统的功能，减轻维护负担等。

2.7.1 注释语句

在程序代码中，适当加入注释可以提高程序的可读性，方便代码的阅读和维护。注释语句的格式如下：

- Rem 注释内容。
- '注释内容。

Rem 注释方式一般只适宜齐头方式注释。如果需要在语句尾部加注释，则需要在 Rem 前面加冒号分割开两条语句，或者使用'注释方式。

注意：本书的所有程序步骤都是采用'注释方式。

2.7.2 InputBox 函数

InputBox 函数用于获取输入，该函数将显示一个输入对话框，用户在其中输入内容后选择确认。该函数的返回值为文本框内容的字符串数据。其语法结构为：

```
返回值=InputBox(Prompt,[title][,default][,xpos,ypos][,helpfile,context])
```

其中各参数的含义如下：

- Prompt：显示在输入对话框中的提示信息，最大长度为 1024 个字符。
- title：可选，输入对话框的标题。省略时，显示应用程序名。
- default：可选，文本框中默认的显示内容。如果省略，则文本框为空。
- xpos(ypos)：可选，数值表达式，指定对话框左（上）边与屏幕左（上）边的水平（垂

直）距离。如果省略，则对话框放置在水平居中（屏幕垂直方向下边大约三分之二）的位置。

- helpfile：可选，字符串表达式，标识识别帮助文件，用该文件为对话框提供上下文相关的帮助。如果使用了该参数，也必须提供 context。
- context：可选，数值表达式，由帮助文件的作者指定给某个帮助主题的帮助上下文编号。如果已提供 context，也必须提供 helpfile。

InputBox 函数无论输入的是数字还是字符，其返回值始终都是字符型。单击【确定】按钮，返回文本框中的内容；单击【取消】按钮，将返回一个零长度字符串。

注意：如果 Prompt 中包含了多行，则可以在各行间使用回车符 Chr(13)、换行符 Chr(10)或回车 Chr(13) & Chr(10)来分隔。

2.7.3　MsgBox 函数

该函数主要使用对话框的形式显示一些简单的错误、警告或提示信息给用户，等待用户的相应操作。其用法有语句与函数两种格式。语句格式如下：

```
MsgBox Prompt [,Buttons] [,title] [,helpfile,context]
```

函数格式如下：

```
返回值=MsgBox(Prompt [,Buttons] [,title] [,helpfile,context])
```

和 InputBox 函数的参数意义基本一致，其中不同的是 Buttons 参数。该参数用来指定显示按钮的数目与样式、图标样式、默认按钮以及消息框的强制响应。参数具体设置如表 2-8~表 2-11 所示。

表 2-8　按钮数目与形式设置表

常　　量	值	说　　明
vbOkOnly	0	只显示【确定】按钮
vbOkCancel	1	显示【确定】及【取消】按钮
vbAbortRetryIgnore	2	显示【异常终止】、【重试】及【忽略】按钮
vbYesNoCancel	3	显示【是】、【否】及【取消】按钮
vbYesNo	4	显示【是】及【否】按钮
vbRetryCancel	5	显示【重试】及【取消】按钮

表 2-9　图标样式设置表

常　　量	值	说　　明
vbCriitical	16	显示 Critical Message 图标
vbQuestion	32	显示 Warning Query 图标
vbExclamation	48	显示 Warning Message 图标
vbInformation	64	显示 Information Message 图标

表 2-10　默认按钮设置表

常　　量	值	说　　明
vbDefaultButton1	0	以第一个按钮为默认按钮
vbDefaultButton2	256	以第二个按钮为默认按钮
vbDefaultButton3	512	以第三个按钮为默认按钮
vbDefaultButton4	768	以第四个按钮为默认按钮

表 2-11　强制响应性设置表

常　　量	值	说　　明
vbApplicationModal	0	进入该消息框，当前应用程序被暂停
vbSystemModal	4096	进入该消息框，所有应用程序被暂停

以上表中列出的常量都是 VBA 的系统常量，当需要将多个设置结合使用时，可以使用“+”将各个常量连接起来。例如：

```
MsgBox "你显示了确定按钮和消息图标！",vbOkOnly+vbInformation
```

2.8　数　　组

数组就是由一系列具体相同属性的、连续的数据变量组成的集合。数组是相当常见并使用广泛的一种数据结构。同一个数组具有相同的变量名，通过索引值加以区别数组中的各个元素。

2.8.1　了解数组定义及上下界

定义数组的语法如下：

```
Dim  数组名(n) As type
```

该数组包括的数组元素是 n 个。例如：下面定义一包含 10 个元素的字符串数组。

```
Dim strName(9) As String
```

在默认情况下，数组的元素的索引值是从零开始的。上面的数组在引用其元素时只能引用 strName(0)到 strName(9)之间的 10 个。如果需要索引值从 1 开始计数，需要使用以下命令：

```
Option Base 1
```

当使用该命令后，上面的实例包含的数组元素个数为 9 个，引用时只能使用 strName(1)到 strName(9)间的所有数组元素。

不使用该命令，而又要索引值从 1 开始时，在定义数组时需要采用如下方式：

```
Dim FirstArray(1 to 25) As Integer
Dim SecondArray(20 to 100) As Integer
```

这里定义时明确指示了数组的上下界。每个数组都有一个下界和上界。下界即数组的最小索引值；上界即数组最大索引值。这里的上下界可以任意定义，但是要保证下界小于上界。

2.8.2　多维数组

以上所涉及的数组都是一维数组，一维数组都只需要一个数字就可以确定数组元素在数组中的位置。在某些情况下，只有一个维度是不够的，此时就需要使用多维数组。要声明二维数组，只需要在一维数组的基础上再添加一个参数即可。下面的实例将创建一个包含 5 行和 10 列的数组，共包括 50 个数组元素。

```
Dim MultiArray(1 to 5,1 to 10)
```

2.8.3　动态数组

有时候，可能并不知道到底需要将数组定义多大规模。随着程序的执行，才能确定该数组包含多少个数组元素。虽然可以定义一个很大容量的数组，但是这样做非常浪费系统的内存资源，这时就需要使用动态数组。

动态数组指没有设置大小的数组。声明该数组时，括号内部不用填入任何数值。例如：

```
Dim DynamicArray()
```

当在程序执行过程中，需要修改数组大小时，可以使用 Redim 命令。例如：

```
Redim DynamicArray(1 to 20)
```

其中的上下界可以使用变量，例如下例将活动工作簿中的所有表名存储到数组中。

```
Sub SaveSheetName()
Dim strSheetName() As String
Dim IntSheetsCount As Integer,IntSheetsNumber As Integer

IntSheetsNumber=ActiveWorkbook.Worksheets.Count          '获得活动工作簿工作表的数目
Redim strSheetName(1 to IntSheetsNumber)                 '重定义数组大小
For IntSheetsCount=1 To IntSheetsNumber                  '循环工作表，将表名存储到数组
    strSheetName(IntSheetsCount)=ActiveWorkbook.Sheets(IntSheetsCount).Name
Next
End Sub
```

有时候，可能需要反复修改数组的大小。但是每次使用 Redim 后，数组都会被重新初始化，其中存储在数组中的数据都会丢失。为了避免出现这种情况，需要使用 Preserve 命令。下面的实例将 C 盘根目录下所有 TXT 文件的文件名放入一个数组。

```
Sub SavetxtFileName()
Dim strFileName As String, arrName() As String
Dim IntFileCount As Integer
```

```
strFileName=Dir("C:\*.txt")                              '取得第一个 C 盘根目录下的 TXT 文件名称
Do Until strFileName=" "                                 '当文件名称获取成功时，执行循环体
     IntFileCount= IntFileCount+1                        '将文件计数器累加一次
     Redim Preserve arrName(1 To IntFileCount)           '修改文件名数组的大小
     arrName(IntFileCount)= strFileName                  '将获得的文件名保存到文件名数组中
     strFileName=Dir                                     '继续获取下一 TXT 文件的文件名
Loop
End Sub
```

2.8.4　5 个数组相关函数和语句

本节将介绍 5 个与数组相关的函数与语句，分别是 Array、IsArray、Erase、LBound 和 Ubound。

1．Array

该函数返回一个 Variant 数据类型的数组。其语法格式如下：

```
数组名=Array(参数列表)
```

参数列表是用逗号隔开的值表，用于为数组的各元素赋值。如果不提供参数，则创建一个零长度的数组。

```
Dim arr As Variant
arr = Array(10,20,30)
```

2．IsArray

该函数返回一个布尔值，标识变量是否为一个数组。其语法格式如下：

```
IsArray(变量名或数组名)
```

如下是该函数使用的一个实例：

```
Dim MyArray(1 To 5) As Integer, YourArray, MyCheck       '声明数组变量
YourArray = Array(1, 2, 3)                               '使用数组函数
MyCheck = IsArray(MyArray)                               '返回 True
MyCheck = IsArray(YourArray)                             '返回 True
```

3．Erase

该语句用于重新初始化大小固定的数组的元素，以及释放动态数组的存储空间。格式如下：

```
Erase 数组名
```

下面是该函数应用的一个实例：

```
Dim NumArray(10) As Integer                              'Integer 数组
Dim StrVarArray(10) As String                            '变长的 String 数组
Dim VarArray(10) As Variant                              'Variant 数组
```

```
Erase NumArray                                  '将每个元素设为 0
Erase StrVarArray                               '将每个元素设为零长度字符串
Erase VarArray                                  '将每个元素设为 Empty
```

4．LBound 和 UBound

LBound 返回一个 Long 型数据，其值指定数组下界。UBound 返回一个 Long 型数据，其值指定数组上界。

2.8.5 在 VBA 中使用数组

1．使用连续单元格区域的值填充数组

用 Excel 做 VBA 开发时，经常需要获取接连的几行几列单元格的值。如果循环各个单元格来获取值，速度比较慢，而且需要多个步骤才能完成，这时可以使用数组，将这些连续的单元格直接填充到该数组中。通常有两种方法，实例如下：

```
arrFill=ActiveWorkbook.sheet1.Range(“A1:C5”)
arrFill= ActiveWorkbook.sheet1.Range(“DataArea”)
```

此处第二种方式，预先定义了连续单元格的名称，故可以使用名称直接访问单元格区域。

2．数组的传递

在 VBA 中有时候需要传递多个值到其他的函数或过程做进一步处理。如果这些值的类型以及使用方式都一致时，可以考虑使用数组进行传递，使代码运行更加快速和更易阅读。下面的实例中子过程 CheckValue 将数组传递到函数 Less5Number 中，用于计算 A1:C3 单元格范围内小于 5 的单元格个数。

```
Sub CheckValue()
Dim arr() As Variant

arr = Sheet1.Range("A1:C3")                  '将 A1:C3 单元格的数据填充到数组 arr 中
MsgBox "A1:C3 单元格范围内共有" & Less5Number(arr) & "个单元格的数据小于 5！"
End Sub

Function Less5Number(ByRef arrData As Variant) As Integer
Dim i, j, intCount As Integer

For i = 1 To 3
    For j = 1 To 3
        If arrData(i, j) < 5 Then
            intCount = intCount + 1
        End If
    Next
Next

Less5Number = intCount
End Function
```

第 3 章　Excel 2007 VBA 对象模型

Excel 2007 VBA 对象模型用于描述 Excel 2007 各对象之间的关系。在 Excel 2007 中使用 VBA 进行应用开发时，首先清楚了解 Excel 2007 的对象模型可以起到事半功倍的效果。Excel 所包含的对象非常多，但是其中很多并不常用。本章重点介绍在 VBA 开发中常用的对象以及其属性和方法。这些对象包括 Application 对象、Workbook 对象、Worksheet 对象和 Range 对象。这 4 个常用对象是按照层级结构组织的（如图 3-1 所示）。

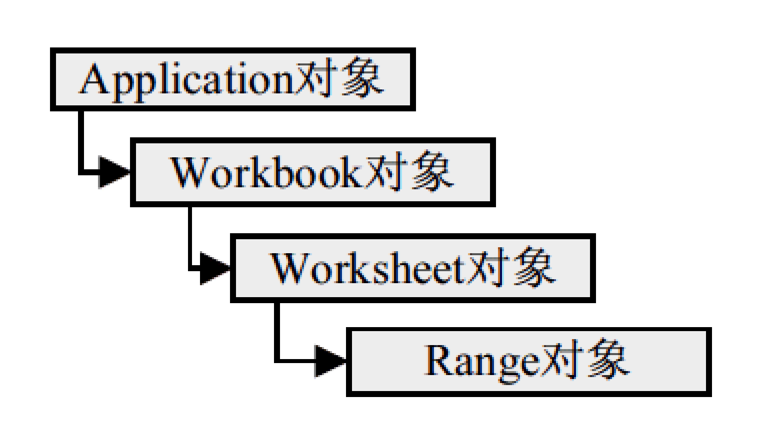


图 3-1　Excel 2007 VBA 对象模型

3.1　面向对象编程

“对象”是面向对象程序设计的核心，明确这个概念对理解面向对象程序设计来说至关重要。所谓对象就是具有某些特性的具体事物的抽象。在现实生活中，其实大家随时随地都在和对象打交道，比如骑的自行车、看的书甚至于单个的人，都是可以被视为对象。

这些现实生活中的对象有 3 个共同的特点。第一，对象都有自己的状态。例如一个球有自己的质地、颜色、大小；第二，对象都具有自己的行为，例如一个球可以滚动、停止或旋转；第三，对象都具有激发其行为的特殊事件，比如要让一个球滚动，需要给这个球的表面施加一个外力，而要球停止，需要有阻力施加到球上。

在面向对象的程序设计中，对象的概念就是对现实世界中对象的模型化。在面向对象编程中，程序通过定义一部分变量并赋予其具有实际意义的值作为对象固有属性的描述，还通过建立一些过程完成具有实际意义的动作来代表对象相应的行为，从而在程序设计中模拟出该对象。这些模拟实际对象属性的变量被称为对象的属性，用于模拟实际对象行为的过程被称为对象的方法。

3.1.1　对象的属性

属性决定一个对象的外观和状态，要改变一个对象的外观和状态，可以通过改变对象的属性实现。在 Excel 2007 VBA 中建立的大多数对象的属性是在对象生成时自动设置的。用户也可以在设计时通过属性窗口或运行时通过代码修改。在运行时可以设置并获得值的属性是可读写属性；在运行时只能读取的属性叫做只读属性。

1．设置属性值

设置属性值可以通过 VBE 开发环境下的属性窗口进行，也可以在程序代码中设置。通过代码设置属性值的格式如下：

```
对象.属性名称=表达式
```

下面的实例将“实例表”的标签名称修改为“标签名修改”。

```
Sheets("实例表")="标签名修改"
```

2．读取属性值

在程序设计过程中，有时候需要将对象当前的属性值保存起来，以便于以后恢复，此时就需要读取属性。读取属性的语法格式如下：

```
变量名=对象.属性名称
```

如果在代码中不是反复使用到对象的该属性，也可以不将该属性的值保存到变量中。下面的实例代码在 Label1 标签控件的显示文字后面添加字符串“（显示字符串）”。

```
Label1.Caption= Label1.Caption & "（显示字符串）"
```

3.1.2　对象的方法

对象的方法是指对象可以进行的操作。方法可以改变对象的属性值，也可以对存储在对象中的数据进行某项操作。例如：对于一个单元格区域对象 Range，可以使用 Select 方法选中某个单元格，也可以使用 ClearContent 方法清除该单元格的内容。对象的方法实际上是对象的一些成员函数。

调用对象的方法时，需要使用点号操作符。如果有参数，则在方法后面加上参数值，参数间用空格隔开。调用的格式如下；

```
对象.方法名称  <参数列表>
```

下面是一个实例，首先选中工作表 Sheet1 的 A1 单元格，然后清除该单元格的内容。

```
With Sheet1.Range("A1")
    .Select                                  '选择 A1 单元格
    .ClearContent                            '清除 A1 单元格的内容
End With
```

3.1.3　对象的事件

事件是对象识别的需要响应的某些用户行为和动作。很多情况下，Excel 2007 VBA 事件是通过用户的交互操作发生的。可以激发事件的用户动作包括切换工作表、选择单元格、单击鼠标、双击鼠标等。当事件发生时，将执行包含在事件过程中的代码。

事件按照对象层次结构分为应用程序事件、工作簿事件、工作表事件和图表事件。每一

种层次都包含很多种事件。在 VBE 编辑器中，当打开了对应对象的代码编辑窗口后，在该窗口右上方的下拉列表框中可以选择对应对象的事件。

有时候，事件会触发其他事件，包括该事件本身。例如：当单元格中的内容发生变更时，工作表的 Worksheet_Change 事件被触发。如果在 Worksheet_Change 事件的过程代码中又修改了一个单元格内容，那么事件将被再次激发。以此类推，循环往复，过程就陷入了死循环之中。为了防止该情况发生，可以在事件过程开始部分设置禁止激发事件，最后在过程的尾部重新启动。禁止事件激发的方法如下例所示：

```
Private Sub Worksheet_Change(Byval Target As Range)
    Application.EnableEvents=False                    '禁止激发事件
    Range("A1").Value=Target.Value                    '修改 A1 单元格的值
    Application.EnableEvents=True                     '取消禁止激发事件
End Sub
```

注意： 当程序陷入死循环后，可以按 Esc 键或 Ctrl+Break 组合键中断程序执行。

3.2　Application 对象

Application 对象处于 Excel 2007 VBA 对象层级结构的最高层。该对象代表了当前正在运行的 Excel 2007 应用程序，其他的对象都是该对象的子对象。设置 Application 的属性将影响整个 Excel 2007 应用程序。

3.2.1　Application 对象常用属性

Application 对象的属性很多，以下给出的只是其中常用的属性。

- ActiveWorkbook：返回当前活动的工作簿。
- ActiveSheet：返回当前活动的工作簿中活动的工作表。返回的工作表可以是 Excel 支持的所有工作表类型，包括工作表或图表。
- ActiveCell：返回当前活动工作簿中活动工作表中的活动单元格。
- AskToUpdateLinks：设置当打开带有链接的文件时询问用户是否更新链接。设置为 True 时，Excel 显示对话框询问是否更新链接；设置为 False 时，Excel 自动更新链接且不显示对话框。
- Caption：返回或设置 Excel 程序标题栏上的标题。
- DisplayAlerts：设置是否显示警告信息对话框。设置为 True 时，显示警告信息；设置为 False 时，不显示警告信息。
- DisplayStatusBar：设置 Excel 状态条是否显示。设置为 True 时，显示状态条；设置为 False 时，不显示状态条。
- DisplayFormulaBar：设置是否显示编辑栏。设置为 True 时，显示编辑栏；设置为 False 时，不显示编辑栏。

- FileDialog：该属性可以用来开启多种关于文件的对话框。下例是开启打开文件对话框的代码。

```
Application.FileDialog( msoFileDialogOpen).Show
```

- Path：返回应用程序完整路径的字符串。Path 不包括末尾的分隔符和应用程序名称。
- SceenUpdating：开启或关闭屏幕刷新。设置为 True 时，开启刷新；设置为 False 时关闭刷新。
- Selection：确定当前活动的对象是什么。可以是工作表、图表、单元格、图形对象等。

3.2.2　Application 对象常用方法

Application 对象的常用方法如下：

- FindFile：显示【打开】对话框并让用户打开一个文件。如果成功打开一个新文件，则该方法返回 True；如果用户退出该对话框，则该方法返回 False。
- GetOpenFilename：显示标准的【打开】对话框，并获取用户文件名，但不真正打开该文件。
- GetSaveAsFilename：显示标准的【另存为】对话框，获取用户文件名，但不真正保存文件。
- OnKey：当按特定键或特定的组合键时运行指定的过程。
- OnTime：在将来的特定时间运行一个既定过程（既可以是具体指定的某个时间，也可以是指定的一段时间之后）。
- Quit：退出 Excel 应用程序。
- SendKeys：将击键发送给活动应用程序。
- Volatile：用于将用户自定义函数标记为易失性函数。无论何时在工作表的任意单元格中进行计算时，易失性函数都必须重新进行计算。非易失性函数只在输入变量改变时才重新计算。

3.3　Workbook 对象

Workbook 工作簿对象位于 Application 对象的下一层次。一个工作簿对象代表一个 Excel 文件。Worksheet 工作表对象、Range 单元格对象等都包含在该对象中。Workbooks 是该对象的集合，使用 Application 对象的 Workbooks 属性可以访问当前打开的所有工作簿对象。

3.3.1　Workbook 对象常用属性

Workbook 对象常用属性如下：

- ActiveSheet：返回活动工作簿中或指定的窗口或工作簿中的活动工作表。如果没有活

动的工作表，则返回 Nothing。

- FullName：返回 Workbook 对象的名称。以字符串表示，包括其磁盘路径。
- Path：返回一个 String 值，该属性代表应用程序的完整路径，不包括末尾的分隔符和应用程序名称。
- Saved：如果指定工作簿从上次保存至今未发生过更改，则该属性值为 True，否则为 False。该属性可以读写。
- Sheets：返回一个 Sheets 集合，该属性代表指定工作簿中的所有工作表。
- Windows：返回一个 Windows 集合，该属性代表指定工作簿中的所有窗口。

3.3.2 Workbook 对象常用方法

Workbook 对象常用方法如下：

- Activate：激活与工作簿相关的第一个窗口。
- Close：关闭对象。
- PrintOut：打印对象。
- PrintPreview：按对象打印后的外观效果显示对象的预览。
- Protect：保护工作簿使其不被修改。
- Save：保存对指定工作簿所做的更改。
- SaveAs：在另一不同文件中保存对工作簿所做的更改。
- Unprotect：取消工作表或工作簿的保护。如果工作表或工作簿不是受保护的，则此方法不起作用。
- UpdateLink：更新链接、DDE 链接或 OLE 链接。

3.4 Worksheet 对象

Worksheet 对象代表 Excel 工作簿中包含的工作表。通过该对象，可以在程序中完成各种对工作表的操作。多个 Worksheet 对象组成 Worksheets 集合，可以使用该集合集中访问工作表对象。

3.4.1 Worksheet 对象常用属性

Worksheet 对象常用属性如下：

- AutoFilter：如果筛选已打开，则返回一个 AutoFilter 对象。
- AutoFilterMode：如果当前在工作表上显示有“自动筛选”下拉箭头，则该值为 True。本属性与 FilterMode 属性互相独立，可读写。
- Cells：返回一个 Range 对象，该属性代表工作表中的所有单元格。

- CodeName：返回对象的代码名。
- Columns：返回一个 Range 对象。该属性代表活动工作表中的所有列，是列的集合，使用该属性可以访问具体的列。
- FilterMode：如果工作表处于筛选模式，则为 True，否则为 False。
- Hyperlinks：返回 Hyperlinks 集合，该属性代表区域的所有超链接。
- Name：返回或设置 Worksheet 对象的名称。
- Next：返回代表下一个工作表的 Worksheet 对象。
- PageSetup：返回一个 PageSetup 对象，该属性包含用于指定对象的所有页面设置。
- Previous：返回代表下一个工作表的 Worksheet 对象。
- Range：返回一个 Range 对象。该属性代表一个单元格或单元格区域。
- Rows：返回一个 Range 对象。该属性代表指定工作表中的所有行，是行的集合。使用该属性可以访问具体的行。
- ScrollArea：以 A1 样式的区域引用形式返回或设置允许滚动的区域。用户不能选定滚动区域之外的单元格。
- UsedRange：返回一个 Range 对象，该对象表示指定工作表上所使用的区域。
- Visible：返回或设置一个 XlSheetVisibility 值，该属性确定对象是否可见。

3.4.2　Worksheet 对象常用方法

Worksheet 对象的常用方法如下。

- Activate：使当前工作表成为活动工作表。
- Copy：将工作表复制到工作簿的另一位置。
- Delete：删除对象。
- Move：将工作表移到工作簿中的其他位置。
- Paste：将“剪贴板”中的内容粘贴到工作表上。
- PasteSpecial：以指定格式将剪贴板中的内容粘贴到工作表上。可用本方法从其他应用程序中粘贴数据，或以特定格式粘贴数据。
- PrintOut：打印对象。
- PrintPreview：按对象打印后的外观效果显示对象的预览。
- Protect：保护工作表使其不能被修改。
- SaveAs：将对图表或工作表的更改保存到另一个文件中。
- Select：选择对象。
- ShowAllData：使当前筛选列表的所有行均可见。如果正在使用自动筛选，则本方法将下拉列表框内容改为“（全部）”。
- Unprotect：取消工作表或工作簿的保护。如果工作表或工作簿不是受保护的，则此方法不起作用。

3.5 Range 对象

Range 对象是一单元格区域。这个区域可以是一个单元格、某一行、某一列或任意指定单元格区域。在 VBA 代码中可以通过在 Range 对象中指定单元格的坐标来访问某个具体的单元格。

3.5.1 Range 对象的引用方式

在 Excel VBA 中，有很多种方法实现对 Range 对象的引用。以下是常用的方式。

- 单元格坐标：例如 Range("A1")或 Range("A1:E10")。
- Cells 集合：例如 Cells(4,5)。
- ActiveCell：访问当前活动单元格。
- Selection：访问当前选定的单元格区域。
- 单元格名称：首先命名单元格区域，然后利用该名称访问。例如 Range("数据区域")。

3.5.2 Range 对象常用属性

Range 对象常用属性如下：

- Address：返回单元格区域引用的地址。
- Borders：返回一个 Borders 集合。该属性代表样式或单元格区域（包括定义为条件格式的区域）的边框。
- Characters：返回一个 Characters 对象。该属性代表对象文本内某个区域的字符。使用该对象可为文本字符串内的字符设置格式。
- Column：返回指定区域中第一块的第一列的列号。
- Count：返回 Range 对象中单元格的数量。
- CountLarge：返回 Range 区域中的最大值。
- CurrentRegion：返回当前区域中以空行与空列的组合为边界的区域。
- End：返回一个 Range 对象，该对象代表包含源区域的区域尾端的单元格。等同于按组合键 Ctrl+向上键、Ctrl+向下键、Ctrl+向左键或 Ctrl+向右键。
- EntireColumn：返回指定区域的整列（或多列）Range 对象。
- EntireRow：返回指定区域的整行（或多行）Range 对象
- Font：返回一个 Font 对象，该属性代表 Range 对象的字体。
- Formula：返回或设置一个 Variant 值，该属性代表 A1 样式表示法的单元格的公式。
- Interior：返回一个 Interior 对象，该属性代表指定单元格的内部。
- Offset：返回一个 Range 对象，该属性代表位于指定单元格区域的一定的偏移量位置

上的区域。
- Resize：调整指定区域的大小。返回一个 Range 对象，该对象代表调整后的区域。
- Row：返回区域中第一个子区域的第一行的行号。
- Rows：返回一个 Range 对象，该属性代表指定单元格区域中的行。
- Text：返回或设置指定对象中的文本。

3.5.3　Range 对象常用方法

Range 对象常用方法如下：
- Activate：激活单个单元格，该单元格必须处于当前选定区域内。
- AdvancedFilter：基于条件区域从列表中筛选或复制数据。如果初始选定区域为单个单元格，则使用单元格的当前区域。
- AutoFill：对指定区域中的单元格执行自动填充。
- AutoFilter：使用“自动筛选”筛选一个列表。
- AutoFit：更改区域中的列宽或行高以达到最佳匹配。
- Clear：清除整个对象。
- Copy：将单元格区域复制到指定的区域或剪贴板中。
- CopyFromRecordset：将 ADO 或 DAO Recordset 对象中的内容复制到工作表，从指定区域的左上角开始。如果 Recordset 对象包含具有 OLE 对象的字段，则该方法无效。
- Cut：将对象剪切到剪贴板，或者将其粘贴到指定的目的地。
- Delete：删除对象。
- Find：在区域中查找特定信息。
- Merge：由指定的 Range 对象创建合并单元格。
- PasteSpecial：将 Range 对象从剪贴板粘贴到指定的区域中。
- Select：选择对象。
- Sort：对值区域进行排序。
- SpecialCells：返回一个 Range 对象，该对象代表与指定类型和值匹配的所有单元格。
- UnMerge：将合并区域分解为独立的单元格。

以上是部分常用的方法，所涉及的所有对象的属性和方法还有很多，限于篇幅，不再一一列出。

第2篇

简单实例

从本篇开始，将正式讲解 Excel 2007 VBA 开发实例。本篇主要讲解简单的单实例。定义为简单实例，是因为本篇的实例所用到的表、窗体、模块数量并不多。建议读者循序渐进，在进入本篇学习时，每阅读一个实例，可以考虑自己独立开发一个相应实例应用。

第 4 章　客户管理系统

客户资源对于一个企业十分重要，客户资料管理是企业日常管理的重要部分。通常该管理系统被集成在中小型企业管理系统中。本章所介绍的客户管理系统是一个独立的部分，因此并不涉及客户管理与其他企业内部管理相互关联的环节。该实例仅完成客户资料的建立、查询、修改、删除以及打印的任务。内容上有些类似第 8 章的人事管理系统，但本例使用的是窗体，而人事系统则是用 Excel 2007 本身的功能实现的。在开发方式上各有所长，读者可以比较参考阅读。

4.1　系统概述

本实例将客户资料管理从企业管理系统中独立出来，缺少了与企业管理系统中其他功能模块间的相互联系的部分，因而结构比较简单。本实例的功能主要包括客户资料基本管理（建立、查询、修改和删除操作）以及打印工作。本章的重点是窗体开发。

4.1.1　设计思路

本系统是一个独立的、单一的客户资料管理系统，完成最基本的客户资料建立、编辑、删除操作以及打印工作。系统使用了两个工作表：首页工作表与客户表。首页工作表完成跳转任务，客户表存储客户资料信息。在实例的主界面，设计了 3 个跳转按钮，分别完成客户资料管理、客户资料打印以及退出系统功能。

各模块的功能介绍如下：

1．客户资料管理

该模块主要用于完成客户资料的添加、编辑和删除操作。为了便于对记录进行修改，还设计了用于浏览的功能、查询修改和查询删除功能。在该界面下可以很方便地建立客户的资料信息。

2．客户资料查询导出

该模块用于完成对客户资料的查询与导出工作。查询客户资料信息时，可以按照客户的全名称、名称缩写和名称拼音等进行查询。查询获得的结果将会立即显示在 ListView 控件中。需要保存查询结果时，可以单击【输出报表】按钮，该按钮将查询结果重新保存到一个新的工作簿中，完成导出数据功能。

4.1.2　知识点一：显示【开发工具】选项卡

在 Excel VBA 开发时，经常需要从 Excel 2007 操作界面跳转到 VBE 环境。有时候需要在 Excel 2007 工作表中插入 ActiveX 控件或进入编辑模式对工作表中添加的控件进行编辑，此时就需要将开发工具选项卡显示出来。默认情况下，Excel 并没有将开发工具选项卡显示出来。要实现该目的操作步骤如下：

（1）首先选择 Excel 2007 左上角的应用程序图标，此时将会显示 Excel 2007 的系统菜单窗口，如图 4-1 所示。

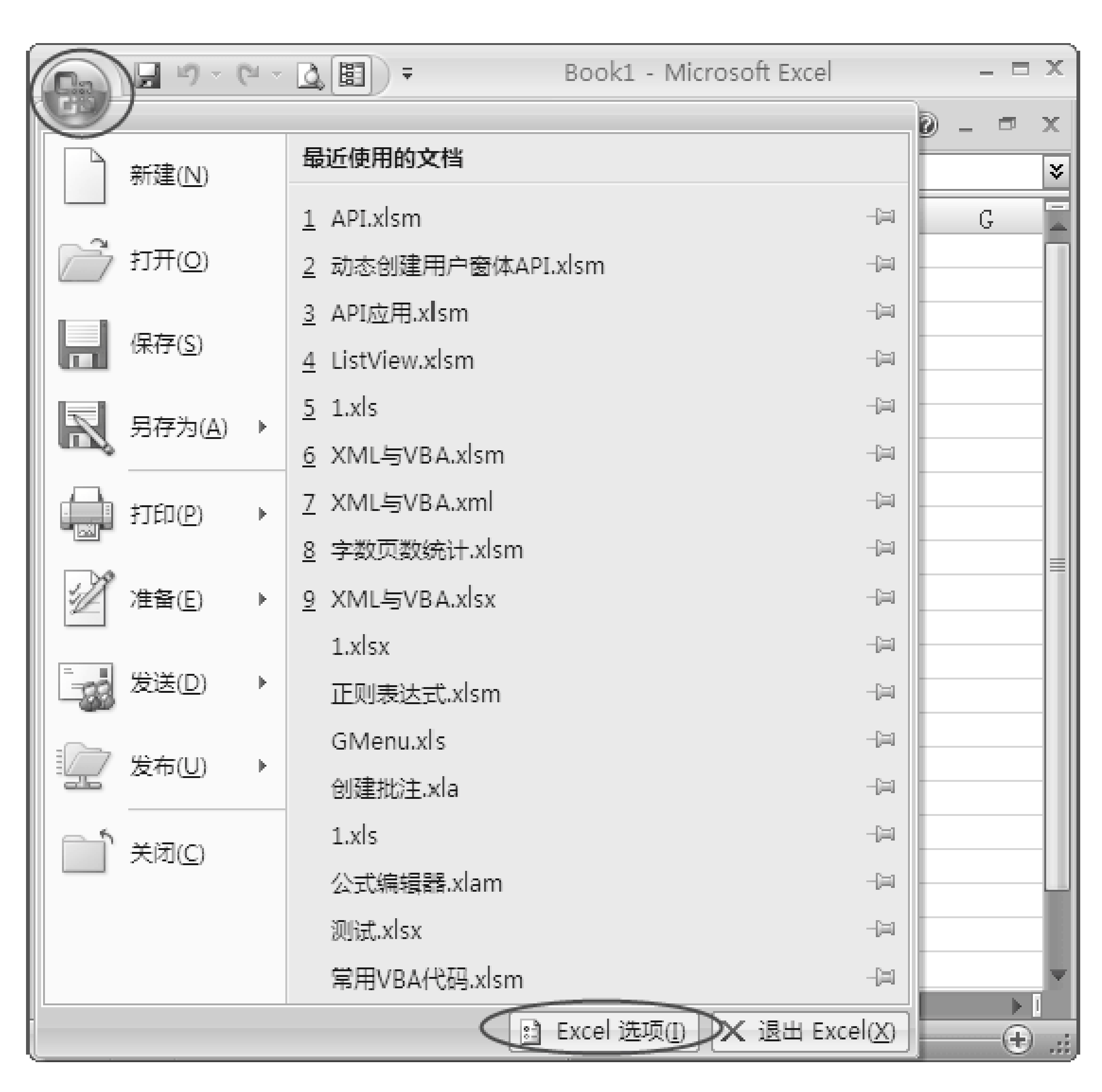


图 4-1　Excel 2007 系统菜单窗口

（2）随后单击【Excel 选项】按钮，将会显示【Excel 选项】对话框，如图 4-2 所示。

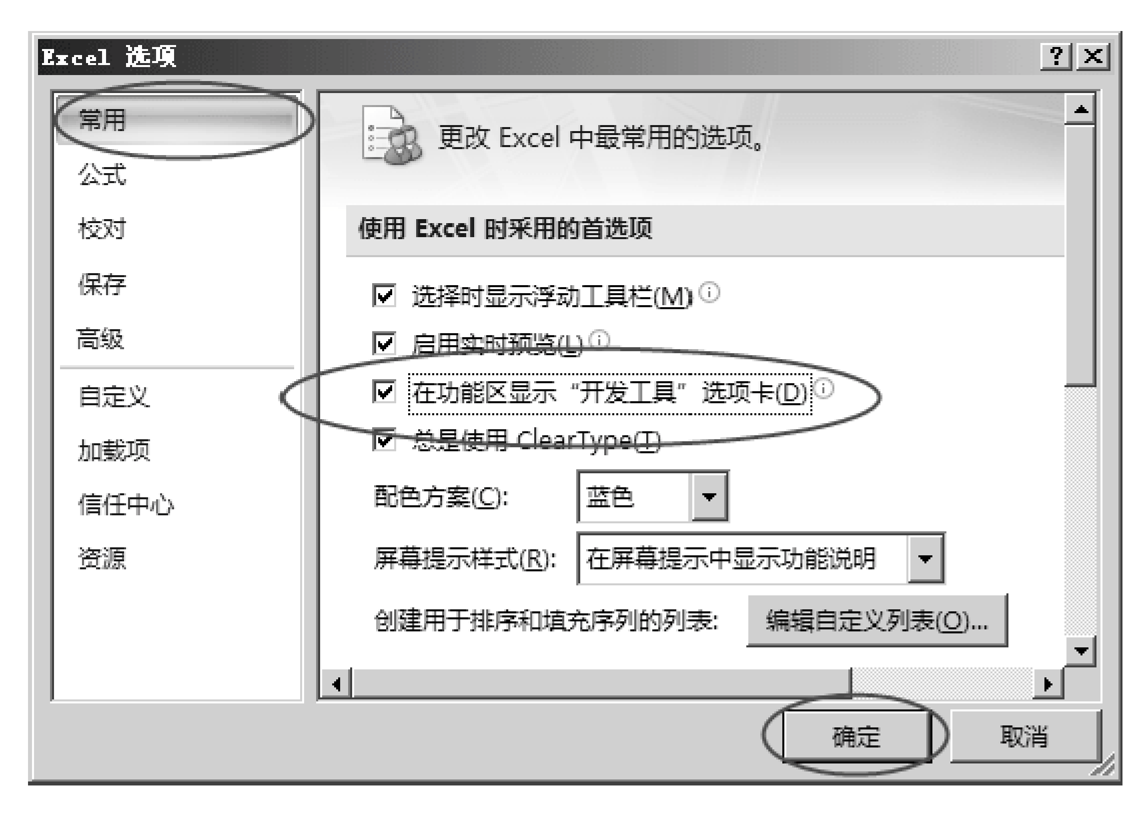


图 4-2　【Excel 选项】对话框

（3）在【Excel 选项】对话框的左侧选择【常用】选项。然后选中右侧的【在功能区显示“开发工具”选项卡】复选框，如图 4-2 所示。最后单击【确认】按钮，此时【开发工具】

选项卡将会出现在 Excel 2007 界面选项卡组中，该选项卡包含的内容如图 4-3 所示。



图 4-3 【开发工具】选项卡

4.1.3 知识点二：开启有代码的工作簿

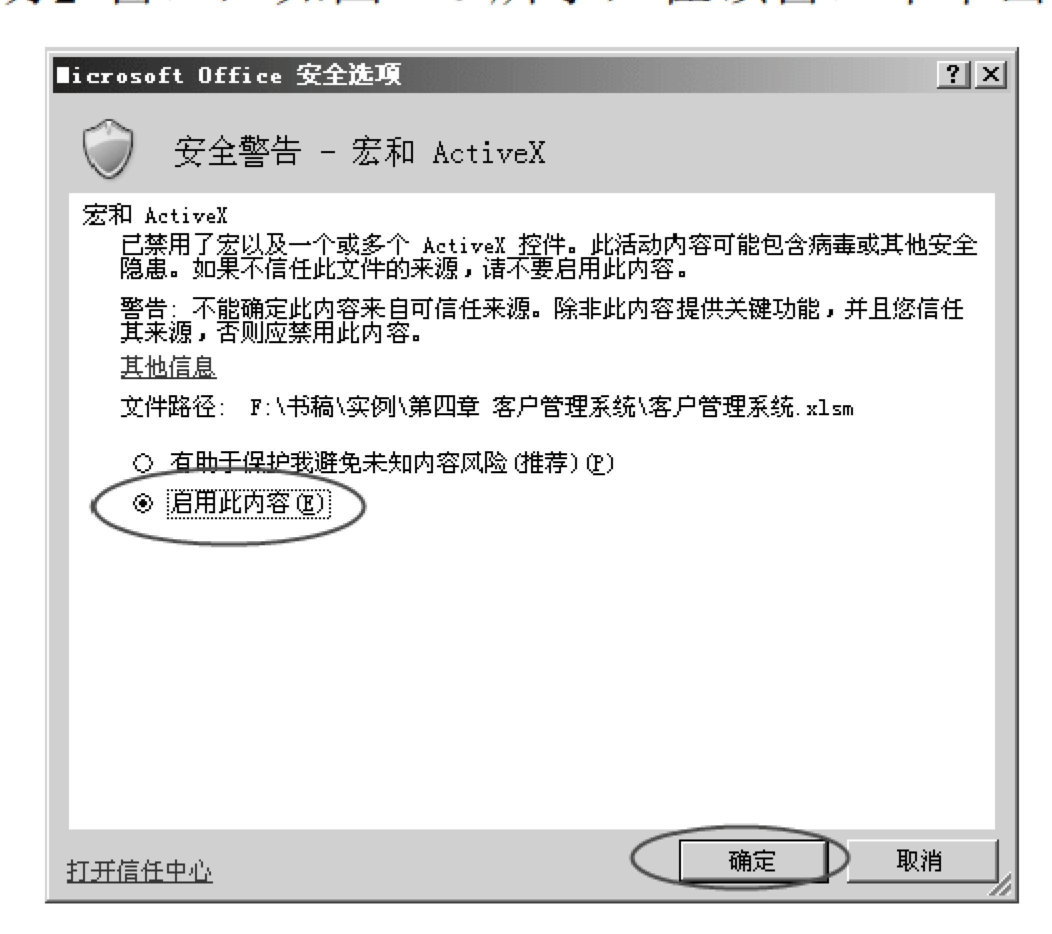
在 Excel 2007 中打开包含 VBA 代码的工作簿时，一般都会询问是否开启工作簿的宏。如果用户需要开启该工作簿中的宏代码，必须选择启用宏选项，否则工作簿中的宏代码将不会发生任何作用。要打开该选项，可以依照以下步骤开启：

（1）按 Ctrl+O 组合键，在随后显示的【打开】对话框中选择任意一个包含 VBA 代码的工作簿文件。本书所有的实例工作簿都是包含宏的工作簿，并保存为 XLSM 格式的工作簿文件。

（2）开启工作簿后，在 Excel 2007 标题栏下方将会显示【安全警告】选项设置栏，如图 4-4 所示。单击【选项】按钮，随后将显示【安全选项】设置窗口，如图 4-5 所示。然后选中【启用此内容】单选按钮，单击【确认】按钮即可。有的工作表在开启时，会直接显示【安全声明】窗口，如图 4-6 所示，在该窗口中单击【启用宏】按钮即可。



图 4-4 安全警告选项设置



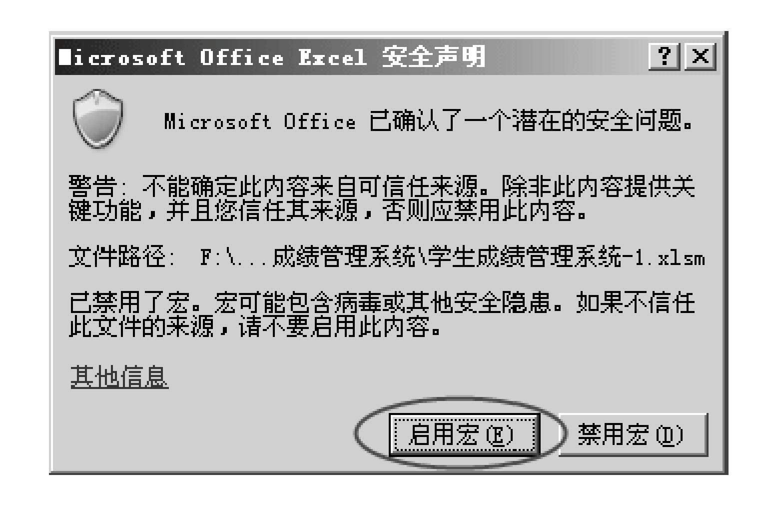


图 4-5 启动宏设置

图 4-6 启用宏

4.2 首 页 设 计

首页工作表是该实例的操作界面，本节介绍该工作表的设计过程。主要设计内容包括界面设计过程、跳转按钮代码与按钮效果代码设计。首页的界面效果如图 4-7 所示。



图 4-7　首页界面效果图

首页所使用到的设计元素详细情况如表 4-1 所示。

表 4-1　首页组成元素列表

名　称	图　示	说　明
人物图片		插入图片，用于首页界面装饰。该图片将被复制两次，调整大小后，重叠放置
标题装饰图片		插入图片，用于点缀首页界面外观。插入在标题前
水平线		直线，用于分割各个功能按钮
圆头竖线		直线，用于对齐各个功能跳转按钮
首页标题	客户管理系统	文本，首页的标题
跳转按钮	客户资料管理	标签窗体控件，用于设计各个跳转按钮

4.2.1　首页界面设计

首页的详细设计过程如下：

（1）插入人物图片。依次在 Excel 2007 中选择【插入】|【图片】命令（如图 4-8 所示）。在打开的【插入图片】对话框中选择“人物图片.bmp”位图文件。然后单击【插入】按钮即可将该图插入首页工作表。该位图可以在“第 4 章”文件目录下找到。

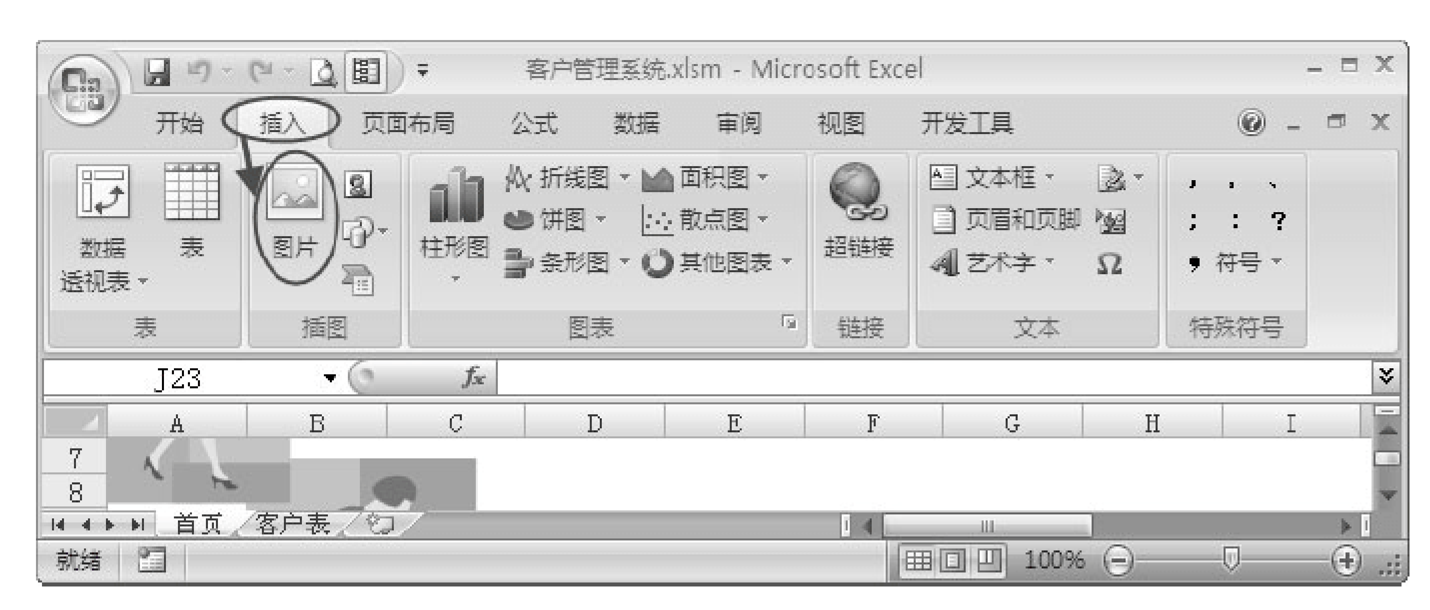


图 4-8　插入图片操作示意图

（2）复制人物图片。单击选中已插入的“人物图片”，随后右击该图片，在弹出的快捷菜单中选择【复制】命令或直接按下 Ctrl+C 键复制该图片。如图 4-9 所示。

（3）在“首页”工作表空白处右击鼠标，在弹出的快捷菜单中选择【粘贴】命令或按 Ctrl+V 组合键粘贴图片到该位置。按照本操作步骤再将“人物图片”复制一份即可。

（4）编辑人物图片格式。右击某一“人物图片”，在弹出的快捷菜单中选择【大小和属性】命令，打开【大小和属性】对话框。将【高度】和【宽度】文本框中的百分比均修改为“47%”，如图 4-10 所示。与次类似，再对其他两“人物图片”的大小与属性进行设置。其设置值如表 4-2 所示。



图 4-9 【图片】快捷菜单

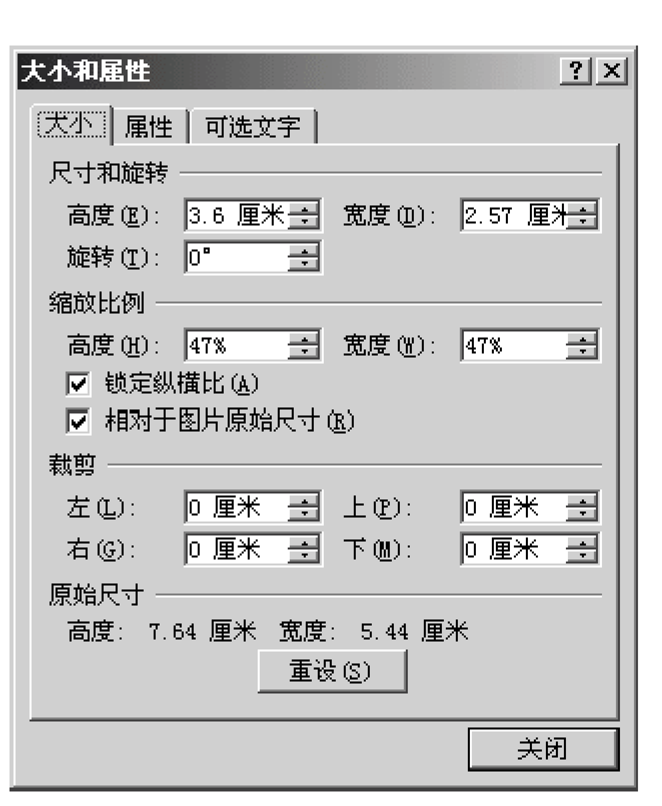


图 4-10 图片的大小与属性设置

表 4-2 “人物图片”的大小与属性缩放比例设置

名　称	高　度	宽　度
第二个“人物图片”	58%	58%
第三个“人物图片”	90%	90%

（5）调整人物图片间位置关系。右击缩放比例为 47%的“人物图片”，在弹出的快捷菜单中选择【置于顶层】命令。然后拖动该图片至 3 个图片的最上方。对缩放比例为 90%的“人物图片”实施类似的操作。只是此时应该选择【置于底层】命令且使其位于 3 个图片的最下方。最终 3 个人物图片的缩放比例及位置关系如图 4-7 所示。

（6）插入标题装饰图。在 Excel 2007 中选择【插入】|【图片】命令，如图 4-8 所示。在随后显示的【插入图片】对话框中选择“装饰图.bmp”位图文件。然后单击【插入】按钮即可将该图插入首页工作表。该位图可以在“第 4 章”文件目录下找到。

（7）添加水平线。在 Excel 2007 中依次选择【插入】|【形状】命令，如图 4-11 所示。随后在【线条】分类栏中选择【直线】选项。在首页工作表中空白处按住鼠标左键不放并水平拖动鼠标，即可产生一水平线。在水平拖动的同时按住 Shift 键可保证产生一绝对水平的直线。依照此操作再次绘制其他两条水平线。



图 4-11　插入形状操作示意图

（8）设置水平线。右击第一条直线，在弹出的快捷菜单中选择【大小和属性】命令，打开【大小和属性】对话框。在【大小和属性】对话框中将尺寸和旋转分类中的【宽度】和【高度】文本框均修改为 28.89 厘米。依照此方法设置另外两条直线的宽度为 10 厘米。

（9）调整水平线位置。单击第一条水平线不动并拖动该水平线。将其起始位置拖动到第一个人物图片的中部、首页标题下方。最终位置如图 4-7 所示。

（10）插入垂直线。在 Excel 2007 中依次选择【插入】|【形状】命令，如图 4-11 所示。随后在【线条】分类栏中选择【直线】选项。在首页工作表中空白处按住鼠标左键不放并垂直拖动鼠标，即可产生一垂直线。在垂直拖动的同时按住 Shift 键可以保证产生绝对垂直线。

（11）设置垂直线格式。右击垂直线，在弹出的快捷菜单中选择【设置形状格式】命令，打开【设置形状格式】对话框。在对话框左侧的选项卡列表中选择【线型】选项卡，如图 4-13 所示。然后在右侧选项设置栏中，选择【箭头设置】的【后端类型】中的【钻石形箭头】样式，如图 4-12 所示。此处只需要尾端呈现钻石形状，具体的效果见表 4-1 中的实例图片。然后复制该垂直线两次即可。

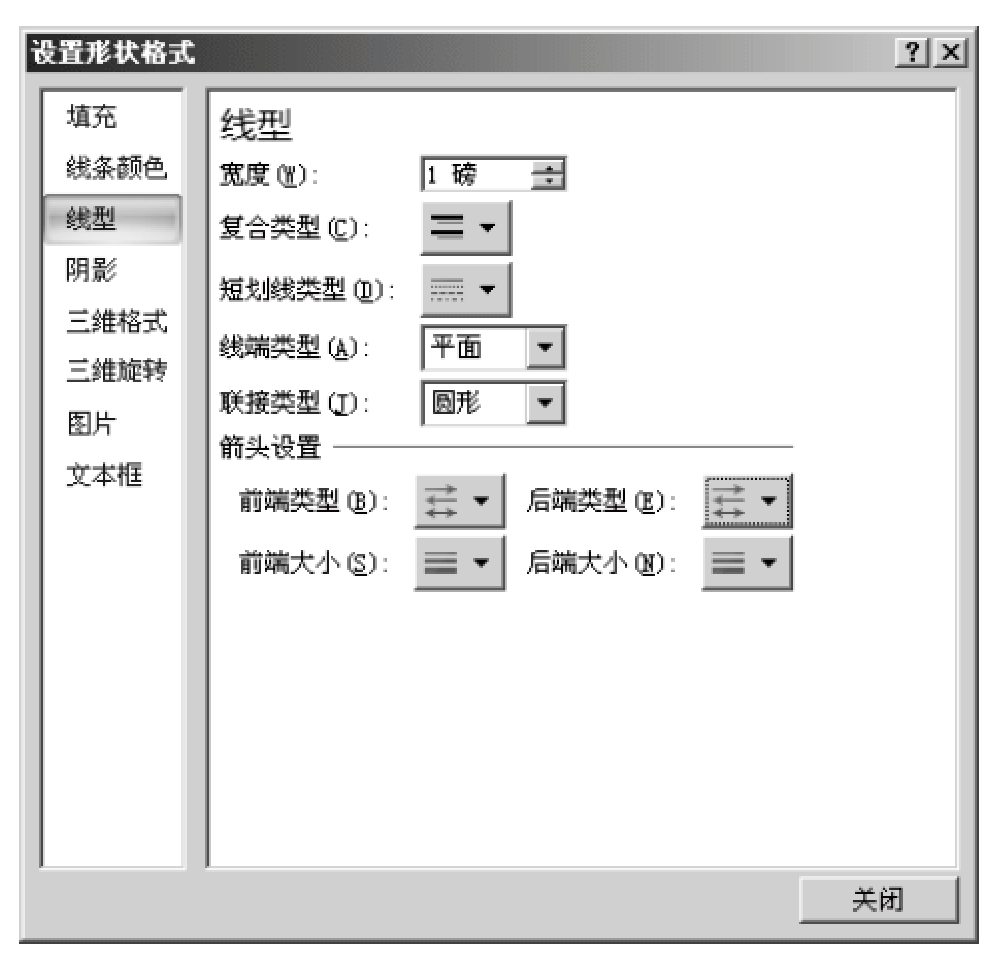


图 4-12　箭头类型设置

图 4-13　设置形状格式窗体

（12）调整垂直线位置。这里只需要将 3 个垂直线的首尾相连即可。首先单击第二个垂

直线不放并拖动，让其头部与第一个垂直线的尾端重合。读者可以使用方向键对移动进行微量调整。与此类似，将第 3 条垂直线的头部与第 2 条垂直线的尾端重合。调整好后，按住 Ctrl 键依次选中所有垂直线，然后将所有垂直线的位置调整成图 4-7 所示效果。

（13）添加标题文本。在 Excel 2007 中依次选择【插入】|【文本框】|【横排文本框】命令，如图 4-14 所示。然后在插入的“装饰图”图片后的空白处单击鼠标左键，产生一文本框。随后在该文本框中输入文字内容“客户管理系统”。

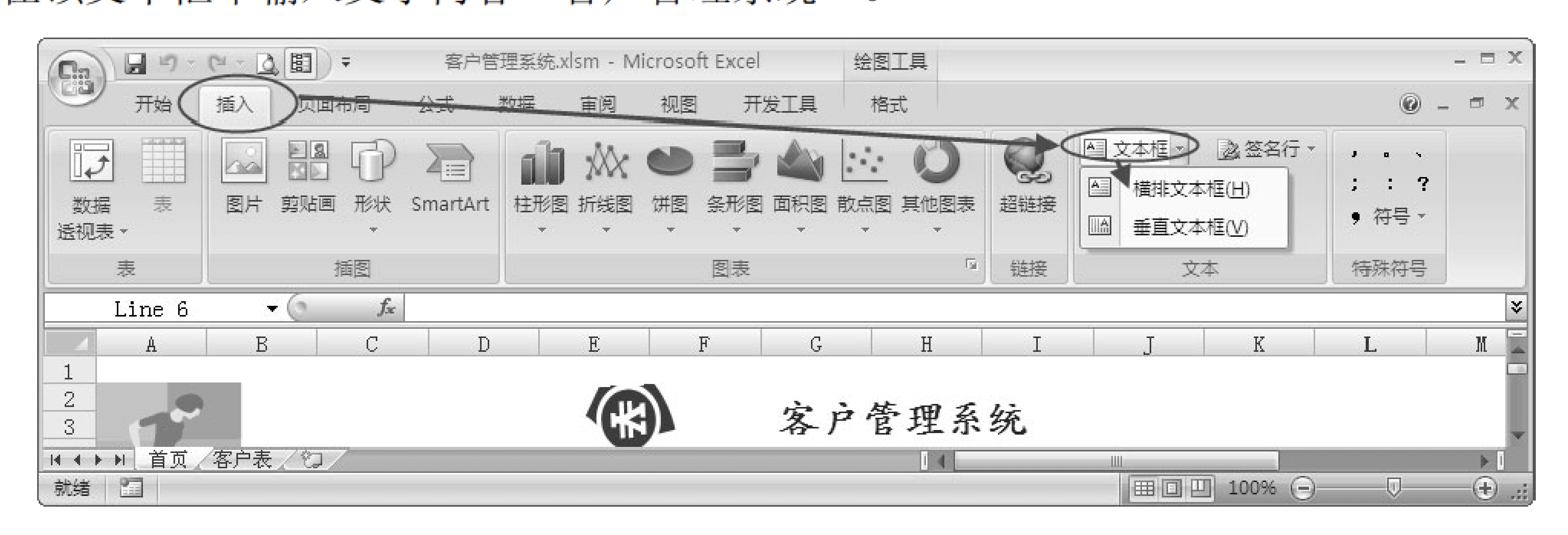


图 4-14　插入文本框操作示意图

（14）设置文本框格式。选中步骤（12）创建文本框中的所有文字内容。此时在选中文字的周围将显示文本格式设置工具栏，如图 4-15 所示。在工具栏中设置其字体为“华文楷体”，字号为 20，加粗，居中对齐格式。最终效果如图 4-7 所示。

（15）创建功能按钮。在 Excel 2007 中依次选择【开发工具】|【控件】|【插入】命令，在 ActiveX 控件列表中选择标签控件，如图 4-16 所示。

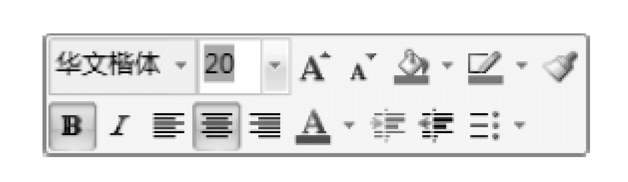


图 4-15　文本设置工具栏



图 4-16　插入控件

（16）设置功能按钮格式。右击刚创建的标签控件，在弹出的快捷菜单中选择【属性】命令，打开【属性】窗口。然后将该标签控件的名称修改为“Label 客户管理”。随后将其 Caption 属性修改为“客户资料管理”。然后选择【按分类序】选项卡将属性按分类排序，如图 4-17 所示。在【杂项】分类中设置该标签控件的 Height 和 Width 属性分别为 26.75 和 175.5。然后通过复制与粘贴命令将该标签控件复制两份。分别将它们的名称修改为“Label 客户资料查询导出”和“Label 退出”。Caption 依次为 “客户资料查询导出”和“退出”。

（17）调整功能按钮的位置。将 3 个按钮依次拖入由垂直线和水平线构成的方框中。可以通过上下左右方向键控制移动的精度。最终效果如图 4-7 所示。



图 4-17 设置客户资料管理标签控件

4.2.2 标签控件显示效果变化代码

首页工作表中的代码其实现的主要功能包括两部分，分别是鼠标滑动时标签控件的显示效果变化以及标签控件的跳转。本小节介绍的是标签控件显示效果变化代码。

该部分代码实现的是一种标签突出显示效果。该代码可以对当前鼠标所指向的标签控件进行突出显示，以达到提示作用。其实现方法是：在 3 个标签控件的 MouseMove 事件过程中，分别执行一个共同的过程 ChangeFace。该过程根据传递的参数，判断具体应该对哪一个标签控件进行操作，然后修改该标签控件的状态。

标签控件的状态变化过程如下：当鼠标在 3 个标签控件间滑动时，鼠标当前指向的标签控件的背景色被修改为&HE0E0E0，其字号被修改为 16。而未被鼠标指向的其他标签控件的显示状态将被复原。该效果的实现代码如下：

```
Option Explicit

'该过程依次检测 3 个标签控件，并修改当前鼠标所指向的标签的显示状态
'当不是指定的标签时，将该标签的显示状态复原
Private Sub ChangeFace(strName As String)
If strName = "客户管理" Then                '检测所操作的标签是否是客户资料管理标签
    '突出显示客户资料管理标签控件
    With Label 客户资料管理
        .Font.Size = 16                     '修改标签控件的字体大小
        .BackColor = &HE0E0E0               '修改标签控件的背景色
    End With
Else
    '复原客户管理标签控件显示状态
    With Label 客户资料管理
        .Font.Size = 14                     '修改标签控件的字体大小
        .BackColor = &H80000005             '修改标签控件的背景色
    End With
End If
If strName = "客户资料查询导出" Then        '检测所操作的标签是否是客户资料查询导出标签
```

```
    '突出显示客户资料查询导出标签控件
    With Label 客户资料查询导出
        .Font.Size = 16                     '修改标签控件的字体大小
        .BackColor = &HE0E0E0               '修改标签控件的背景色
    End With
Else
    '复原客户资料查询导出标签控件显示状态
    With Label 客户资料查询导出
        .Font.Size = 14                     '修改标签控件的字体大小
        .BackColor = &H80000005             '修改标签控件的背景色
    End With
End If
If strName = "退出" Then                    '检测所操作标签是否是退出标签
    '突出显示退出标签控件
    With Label 退出
        .Font.Size = 16                     '修改标签控件的字体大小
        .BackColor = &HE0E0E0               '修改标签控件的背景色
    End With
Else
    '复原退出标签控件显示状态
    With Label 退出
        .Font.Size = 14                     '修改标签控件的字体大小
        .BackColor = &H80000005             '修改标签控件的背景色
    End With
End If
End Sub

'当鼠标在客户资料管理标签上移动时，执行该事件过程
Private Sub Label 客户资料管理_MouseMove(ByVal Button As Integer, ByVal Shift As Integer, ByVal
X As Single, ByVal Y As Single)
ChangeFace "客户资料管理"          '调用 ChangeFace 过程，改变客户资料管理标签的显示状态
End Sub

'当鼠标在客户资料查询导出标签上移动时，执行该事件过程
Private Sub Label 客户资料查询导出_MouseMove(ByVal Button As Integer, ByVal Shift As Integer,
ByVal X As Single, ByVal Y As Single)
ChangeFace "客户资料查询导出"      '调用 ChangeFace 过程，改变客户资料查询导出标签的显示状态
End Sub

'当鼠标在退出标签控件上移动时，执行该事件过程
Private Sub Label 退出_MouseMove(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As
Single, ByVal Y As Single)
ChangeFace "退出"                  '调用 ChangeFace 过程，改变退出标签的显示状态
End Sub
```

4.2.3 标签单击事件代码

该段代码实现首页的跳转功能。当在首页工作表的各个标签上单击时，将执行相应的标

签的 Click 事件过程代码，从而激发相应的功能。

首页中包含了 3 个标签，分别是客户资料管理标签、客户资料查询导出标签和退出标签。单击客户资料管理标签控件将显示客户信息管理窗体；单击客户资料查询导出标签将显示客户资料查询导出窗体；单击退出标签将保存工作簿后退出该工作簿。以下是该代码块的代码解释：

```
Private Sub Label 客户资料管理_Click()
客户信息管理.Show                    '显示客户信息管理窗体
End Sub

Private Sub Label 客户资料查询导出_Click()
客户资料查询导出.Show                '显示客户资料查询导出窗体
End Sub

Private Sub Label 退出_Click()
With ThisWorkbook
    .Save                            '保存当前工作簿
    .Close                           '关闭当前工作簿
End With
End Sub
```

4.3　客户资源管理窗体设计

客户资源管理的功能在客户资源管理窗体中完成。该窗口主要完成客户资料的添加、查找、修改、删除以及记录的浏览工作。使用查找功能可以快速定位需要修改和删除的客户记录。窗体中用于浏览记录的按钮比较多，用户可以选择的浏览方式也可以分为两种：一种是通过【查看客户】表按钮一次性查看所有客户记录；另一种是通过【首条】、【上一条】、【下一条】和【最后】按钮一条一条地在窗体中浏览客户记录。客户信息管理窗体的界面如图 4-18 所示。

图 4-18　客户信息管理窗体

4.3.1 窗体界面设计

该窗体的左侧是客户信息资料显示与编辑区域，右侧是相应的控制按钮，底部是一提醒信息标签。该窗体的 ShowModal 属性被设置为 Flase，该窗体为无模式窗口。整个界面被分割为 4 个部分，以下详细介绍。

1. 客户信息资料

该部分包括 20 个标签控件（用于提示相应的文本框或复合框控件显示内容）、19 个文本框控件（显示当前浏览的记录，在此可以对该记录进行编辑）和 1 个复合框控件。

这 20 个标签控件分别用来提示以下属性：客户 ID、客户名称、名称缩写、负责人、地址、邮政编码、城市、地区、国家、电话、传真、Email、网址、经营范围、客户级别、信用等级、联系人、联系人电话、联系人 Email 和备注。相应的文本框和复合框控件的 Tab 属性被设置为该顺序，并且在客户资料表中各个信息字段也是按照这个顺序排列。

相应的文本框和复合框控件的 SelectionMargin 被设置为 False。当该属性被设置为 False 时，对应的文本框和复合框控件整个文本区域都可以用来显示和保存文本，默认情况下的文本页边距将被取消。

2. 功能按钮区

该区域包括新增、查找、修改、删除和退出 5 个功能按钮。前 4 个完成客户信息资料的编辑操作，最后一个按钮用于退出窗体。

3. 记录浏览按钮区

该区域包括查看客户表、首条、上一条、下一条和最后 5 个按钮。可以通过该区域的按钮在客户表中或直接在窗体中浏览客户信息资料。

4. 提示区

该区域包括两个标签控件，其中一个直接显示“提示：”，表明该部分用于显示提示信息；另一个动态提示当前浏览的记录在客户表中的行数（并非记录数，因为该表还多了一个标题）。

4.3.2 窗体初始化代码

在该窗体被加载的时候，首先需要完成一些初始化工作，以保证窗体显示工作顺利完成。这些工作包括初始化变量、初始化复合框控件、初始化记录显示和初始化浏览按钮的显示状态。窗口初始化的详细流程如图 4-19 所示。

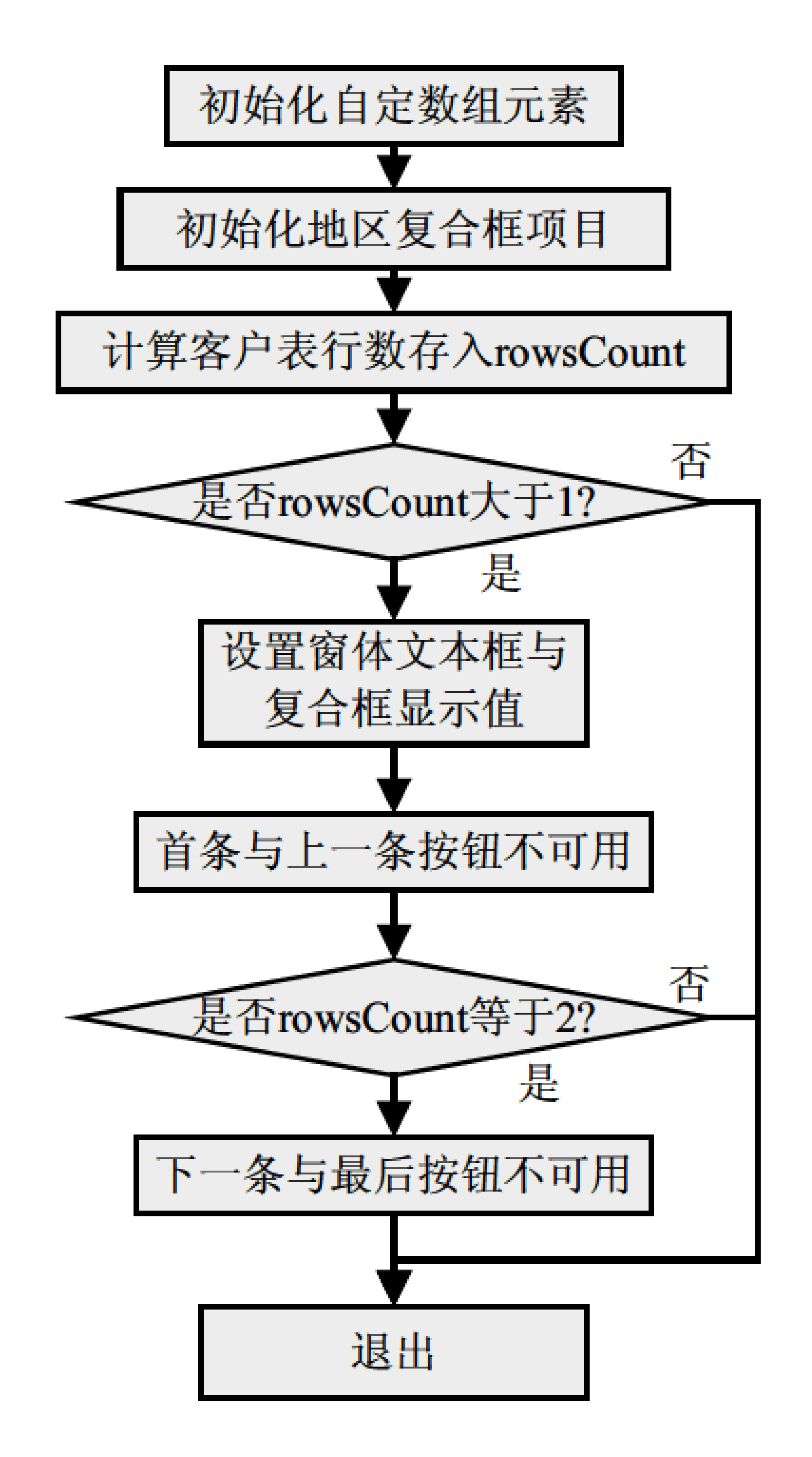


图 4-19　窗口初始化流程图

以下是该过程的代码解释：

```
Private Sub UserForm_Initialize()
    Dim SQL As String
    Dim rowsCount As Integer
    '初始化变量
    Set ws = Worksheets("客户表")                                   '获取客户表工作表对象
    myArray = Array("客户 ID", "客户名称", "名称缩写", "负责人", "地址", _
        "邮政编码", "城市", "地区", "国家", "电话", "传真", "Email", "网址", _
        "经营范围", "客户级别", "信用等级", "联系人", "联系人电话", _
        "联系人 Email", "备注")                                     '初始化数组元素
    '初始化地区复合框项目
    With  地区
        .AddItem "东北"                                             '为地区复合框添加第一个项目
        .AddItem "华北"                                             '为地区复合框添加第二个项目
        .AddItem "西北"                                             '为地区复合框添加第三个项目
        .AddItem "西南"                                             '为地区复合框添加第四个项目
        .AddItem "华东"                                             '为地区复合框添加第五个项目
        .AddItem "华南"                                             '为地区复合框添加第六个项目
        .ListIndex = 0                                              '为地区复合框添加第七个项目
    End With
    '以下是初始化记录显示代码
    rowsCount = Ws.Range("A" & Rows.Count).End(xlUp).Row       '获取客户表的有数据的行数
    '当客户表中存在客户资料记录时，将第一条显示在窗体中
```

```
    If rowsCount > 1 Then
        With Ws
            客户 ID.Text =.Range("A2")                       '设置客户 ID 文本框显示值
            客户名称.Text =.Range("b2")                      '设置客户名称文本框显示值
            名称缩写.Text =.Range("C2")                      '设置名称缩写文本框显示值
            负责人.Text =.Range("d2")                        '设置负责人文本框显示值
            地址.Text =.Range("E2")                          '设置地址文本框显示值
            邮政编码.Text =.Range("f2")                      '设置邮政编码文本框显示值
            城市.Text =.Range("g2")                          '设置城市文本框显示值
            地区.Text =.Range("H2")                          '设置地区复合框显示值
            国家.Text =.Range("i2")                          '设置国家文本框显示值
            电话.Text =.Range("j2")                          '设置电话文本框显示值
            传真.Text =.Range("k2")                          '设置传真文本框显示值
            Email.Text =.Range("l2")                         '设置 Email 文本框显示值
            网址.Text =.Range("m2")                          '设置网址文本框显示值
            经营范围.Text =.Range("n2")                      '设置经营范围文本框显示值
            客户级别.Text =.Range("o2")                      '设置客户级别文本框显示值
            信用等级.Text =.Range("p2")                      '设置信用等级文本框显示值
            联系人.Text =.Range("q2")                        '设置联系人文本框显示值
            联系人电话.Text =.Range("r2")                    '设置联系人电话文本框显示值
            联系人 Email.Text =.Range("s2")                  '设置联系人 Email 文本框显示值
            备注.Text =.Range("t2")                          '设置备注文本框显示值
        End With
        '显示记录为首条时，设置首条和上一条按钮的状态为不可用
        首条.Enabled = False
        上一条.Enabled = False
        '当客户资料表只有一条客户资料时，设置下一条和最后按钮状态不可用
        If rowsCount = 2 Then
            下一条.Enabled = False                           '设置下一条按钮不可用
            最后.Enabled = False                             '设置最后按钮不可用
        End If
        '在提示中显示当前显示记录在客户表中的行数
        提示.Caption = "当前记录在客户表中位于第 2 行"
    Else
        '当客户表中不存在客户资料时，提示无客户资料
        提示.Caption = "当前客户表中还没有任何客户信息记录！"
    End If
End Sub
```

4.3.3　新增按钮代码

在程序运行过程中，【新增】按钮有两种状态。一种是新增状态，该状态下单击【新增】按钮时，将会重置所有文本框和复合框，以便于新建客户资料信息时的输入工作。被选择后该按钮的 Caption 属性将被修改为添加。另一种是添加状态，该状态下单击【新增】按钮将完成添加功能。因而该按钮也包含了对应的两个功能，一个是新建时的重置功能，一个是再次

选择时的添加记录功能。

这两种状态的切换是由用户选择该按钮的次数决定的。当用户选择该按钮的次数为偶数时，按钮处于新增状态，当用户选择该按钮的次数为奇数时，按钮处于添加状态。为了记录用户单击该按钮的次数，该过程中使用了一个 Static 定义的局部静态变量。该过程的流程如图 4-20 所示。

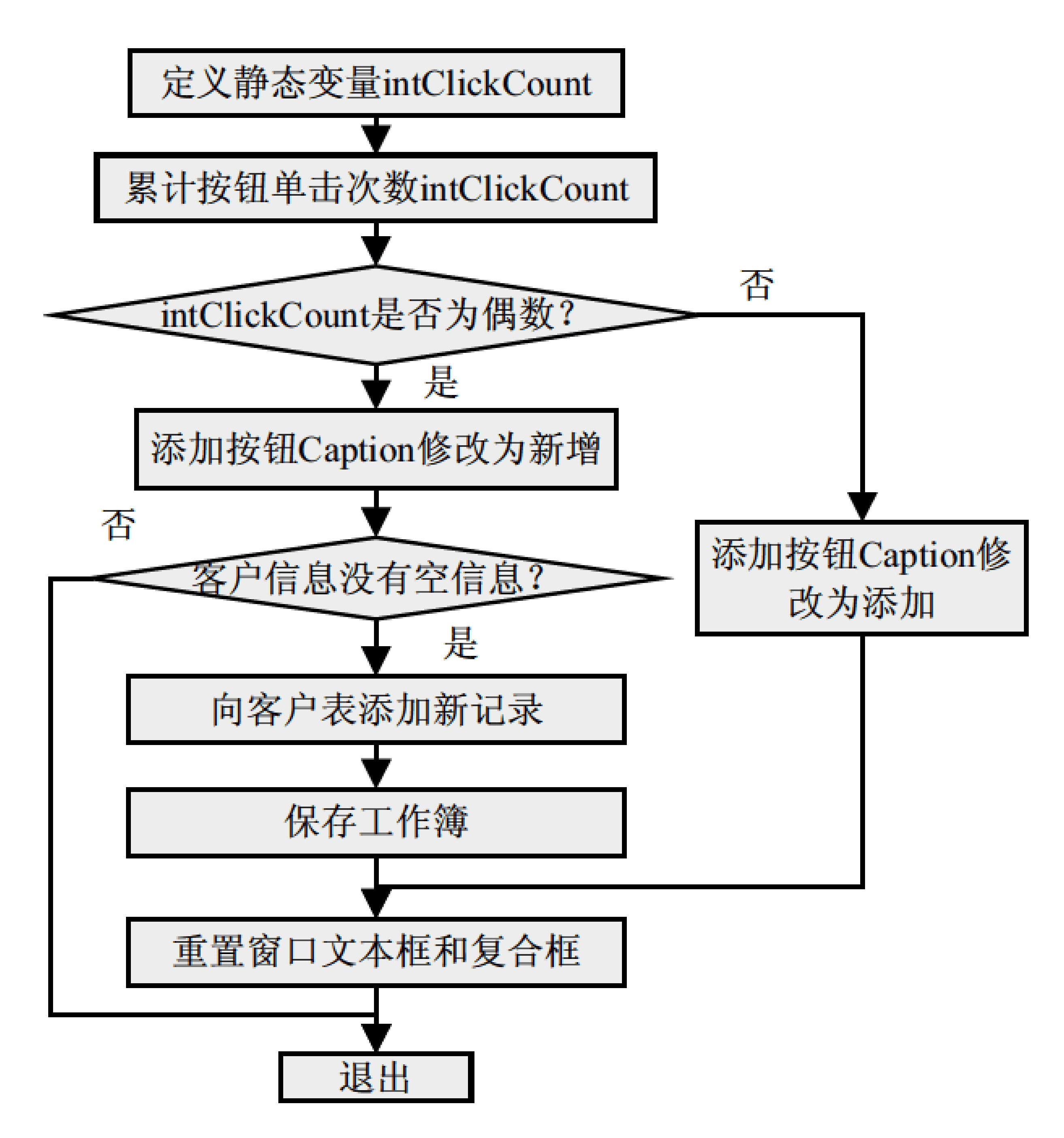


图 4-20　【新增】按钮单击事件过程流程图

该过程的详细代码解释如下：

```
Private Sub 添加_Click()
    Dim n As Long, i As Integer
    Static intClickCount As Integer
    intClickCount = intClickCount + 1
    If intClickCount Mod 2 = 0 Then
        '单击偶数次时，执行添加功能代码
        添加.Caption = "新增"                                  '修改按钮 Caption 属性为“新增”
        '循环检测各个文本框和复合框控件
        For i = 0 To UBound(myArray)
            If myArray(i) = "备注" Then Exit For               '备注项可以为空
            If Me.Controls(myArray(i)).Value = "" Then         '检测其他文本框和复合框是否为空
                MsgBox myArray(i) & "不能为空！", vbCritical, "警告"    '提示项目不能为空
                Me.Controls(myArray(i)).SetFocus               '重置鼠标焦点
                Exit Sub
            End If
        Next
        n = ws.Range("A65536").End(xlUp).Row + 1               '获取新记录在客户表的插入行号
```

```
            For i = 1 To UBound(myArray) + 1
                ws.Cells(n, i) = Me.Controls(myArray(i -1)).Value         '保存新客户的信息到客户表
            Next
            ws.Columns.AutoFit                                            '客户表各列自动适应宽度
            MsgBox "客户信息添加成功！", vbInformation, "添加数据"          '提示添加客户信息成功
            客户 ID.SetFocus                                              '将焦点移到客户 ID 文本框
            地区.ListIndex = 0                                            '设置地区文本框的默认值
            国家.Value = "中国"                                           '设置国家文本框的默认值
            ThisWorkbook.Save                                             '保存工作簿
        Else
            '当是奇数次单击此按钮时，执行以下代码
            添加.Caption = "添加"                          '将按钮的 Caption 属性设置为添加
        End If
        '将窗口中的所有文本框控件和复合框控件重置
        For i = o To UBound(myArray)+1
            Me.Controls(myArray(i)).Value = ""             '重置文本框或复合框
        Next
End Sub
```

4.3.4　查找按钮代码

当用户使用记录浏览按钮浏览客户记录时，只能逐个查看，这样的查看方式不便于快速定位需要修改或删除的客户记录。这时可以就使用【查找】按钮完成该工作。

在本窗口中的【查找】按钮只支持对客户名称的拼音检索，并且一定要保证所输入的拼音是完全正确的。当用户单击该按钮后，将显示一个输入框，要求输入客户名称的汉语拼音字头（如图 4-21 所示）。输入完成后，程序将检索所有客户记录的拼音字头。当有符合条件客户记录时，将在窗体中将该条客户记录显示出来。

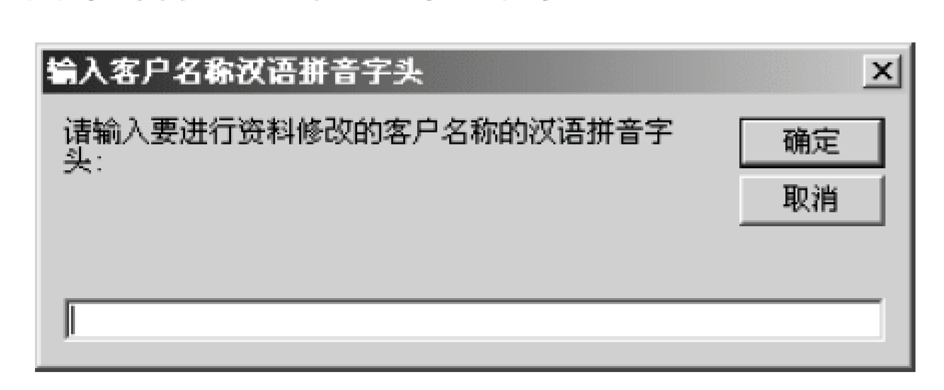


图 4-21　输入客户名称拼音字头

该过程中包含了大量的文本框和复合框数据设置代码，占用了大量的页面，实际上过程流程十分简单。这里不再将该过程的流程表示为图示。下面是单击【查找】按钮的事件代码：

```
Private Sub 查找_Click()
Dim myName As String
        '输入客户名称汉语拼音字头
        myName = InputBox("请输入要进行资料修改的客户名称的汉语拼音字头:", _
            "输入客户名称汉语拼音字头")
        '开始查询某个客户
        '检查拼音函数核对所有客户名称的拼音字头，如果有相符返回真，否则为假
        '该函数的具体使用和代码见本节后续文字
```

```
    If 检查拼音(myName) Then
        '将查询到的客户资料显示在窗体上
        With Ws
            客户 ID.Text = .Range("a" & intRowNow)              '设置客户 ID 文本框显示值
            客户名称.Text = .Range("b" & intRowNow)             '设置客户名称文本框显示值
            名称缩写.Text = .Range("c" & intRowNow)             '设置名称缩写文本框显示值
            负责人.Text = .Range("d" & intRowNow)               '设置负责人文本框显示值
            地址.Text = .Range("E" & intRowNow)                 '设置地址文本框显示值
            邮政编码.Text = .Range("f" & intRowNow)             '设置邮政编码文本框显示值
            城市.Text = .Range("g" & intRowNow)                 '设置城市文本框显示值
            地区.Text = .Range("H" & intRowNow)                 '设置地区复合框显示值
            国家.Text = .Range("i" & intRowNow)                 '设置国家文本框显示值
            电话.Text = .Range("j" & intRowNow)                 '设置电话文本框显示值
            传真.Text = .Range("k" & intRowNow)                 '设置传真文本框显示值
            Email.Text = .Range("l" & intRowNow)                '设置 Email 文本框显示值
            网址.Text = .Range("m" & intRowNow)                 '设置网址文本框显示值
            经营范围.Text = .Range("n" & intRowNow)             '设置经营范围文本框显示值
            客户级别.Text = .Range("o" & intRowNow)             '设置客户级别文本框显示值
            信用等级.Text = .Range("p" & intRowNow)             '设置信用等级文本框显示值
            联系人.Text = .Range("q" & intRowNow)               '设置联系人文本框显示值
            联系人电话.Text = .Range("r" & intRowNow)           '设置联系人电话文本框显示值
            联系人 Email.Text = .Range("s" & intRowNow)         '设置联系人 Email 文本框显示值
            备注.Text = .Range("t" & intRowNow)                 '设置备注文本框显示值
        End With
        浏览按钮状态                                            '修改各个浏览按钮的可用状态
    End If
End Sub
```

代码说明：

- 检查是否有客户名称的拼音字头与所查询的相符，使用检查拼音和拼音头字母两个函数。这两个函数位于该工程文件的公共过程模块中。这两个函数的代码介绍请见本章的后续内容。
- 在该过程中用到了一个模块变量 intRowNow。该变量在窗口代码头部被定义，窗口中所有的过程和函数都可以访问该变量。该变量在检查拼音函数中被修改，用于确定查询到的客户记录所在行数。

4.3.5　检查拼音函数代码设计

检查拼音函数是通过使用一个双重循环，确认在客户表中是否有当前客户名称的拼音字头的客户记录。外层循环检测遍历了客户表中所有客户记录的客户名称。内层循环遍历某一个客户名称中的所有汉字，获得客户名称头字母。最后将这个客户名称的拼音字头与用户输入的拼音字头加以比较，确认该客户记录是否为查询结果。该过程的流程如图 4-22 所示。

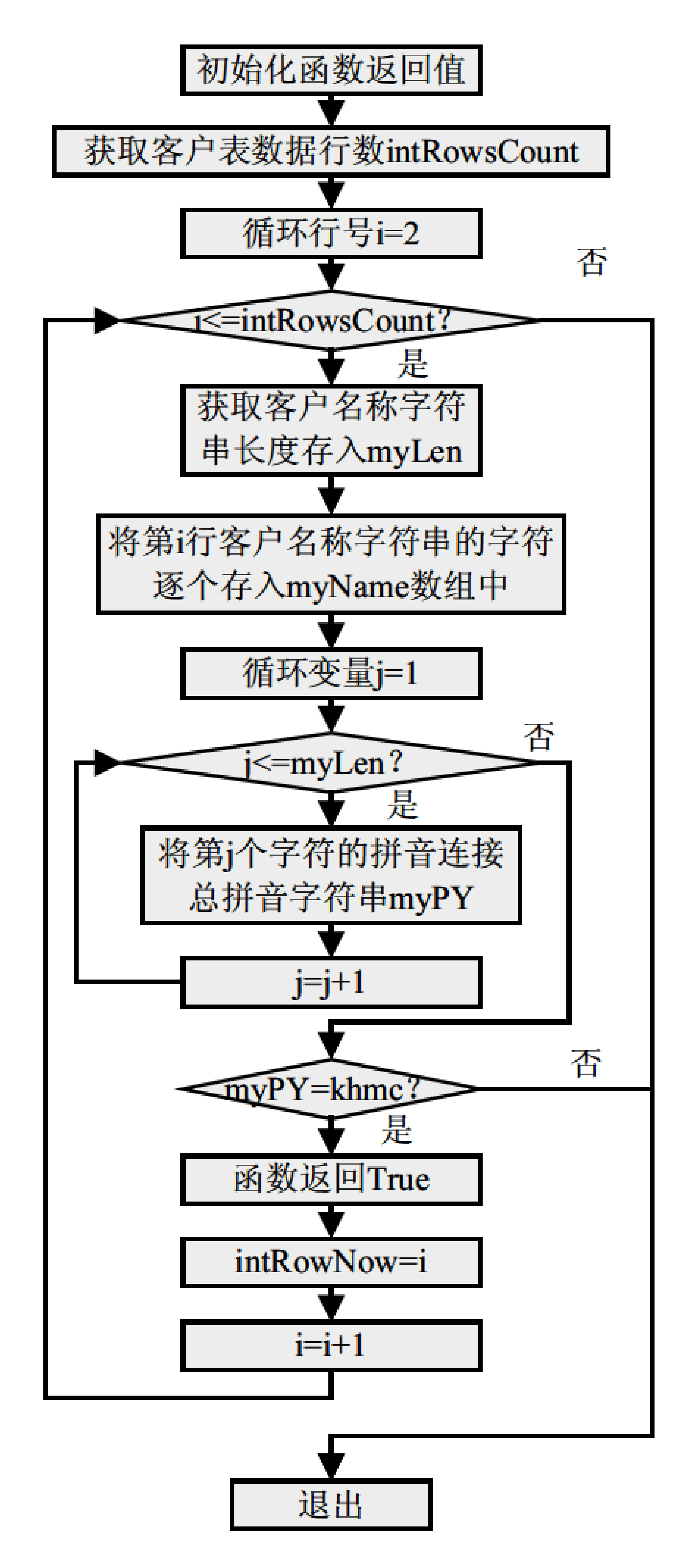


图 4-22　检查拼音函数流程图

以下为该函数的代码：

```
Public Function 检查拼音(khmc As String, intRowNow As Integer) As Boolean
    Dim myLen As Integer, k As Integer
    Dim intRowsCount As Integer
    检查拼音 = False                                    '初始化函数返回值
    '获得客户表有数据的行数
    intRowsCount = Ws.Range("A" & Rows.Count).End(xlUp).Row
    For i = 2 To intRowsCount                           '循环客户表中所有的客户记录行
        '获取每个记录的客户名称字符串的长度
        myLen = Len(Ws.Cells(i, 2))
        '将客户名称的每个汉字逐个保存到数组 myName 中
        ReDim myName(1 To myLen) As String              '重新设定数组上下界
        For j = 1 To myLen
            myName(j) = Mid(Ws.Cells(i, 2), j, 1)       '将单个字符存储在对应数组元素中
        Next
```

```
            '以下代码获取客户名称的汉语拼音的第一个字母
            myPY = ""                                    '重置客户名称字头字符串
            '以下循环遍历汉字数组，获取每个数组元素包含的汉字的拼音首字母并
            '把这些首字母连接起来，保存在 myPY 字符串变量里
            For j = 1 To myLen
                myPY = myPY &  拼音头字母(myName(j))      '将单个字符的拼音字头连接到总字符串
            Next
            '判断是否有复合条件的客户记录
            If LCase(khmc) = LCase(myPY) Then            '使用 Lcase 函数消除大小写的影响
                检查拼音  = True                         '标识结果被查询到
                intRowNow = i                            '记录当前的行号
                Exit Function                            '跳出函数
            End If
        Next
        MsgBox "没有符合条件的客户资料！", vbCritical, "查询结果"  '提示用户没有找到对应的客户信息
End Function
```

4.3.6　拼音头字母函数代码设计

拼音头字母函数接受一个汉字字符参数。函数根据该字符参数，确定出该汉字拼音的头字母，然后将该汉字拼音的头字母作为结果返回。该函数的结构十分简单，但是需要做出的判断比较多，这里不再列出该函数的流程图。

该函数首先获取汉字字符参数的字符代码。由于汉字在字符代码表中是按照拼音的顺序排序的，在确认该汉字的首字母拼音时，只需与字符代码表中各个分段的起始汉字的字符代码比较即可。函数中使用了 If…Then…ElseIf 的结构。

以下是该函数的详细代码设计：

```
Public Function 拼音头字母(myChar As String) As String
    Dim i As Long
    i = Asc(myChar)                                     '获取汉字参数的字符代码
    '以下代码获取该汉字参数的字符代码对应的拼音字母。
    '汉字的字符代码的编排按照拼音字母顺序，所以只需要比较每个拼音字母的
    '首尾汉字的字符代码即可得到对应的拼音字母。
    If i >= Asc("啊") And i < Asc("芭") Then            '检测 i 是否落在 A 首尾汉字拼音字母间
        拼音头字母  = "A"
    ElseIf i >= Asc("芭") And i < Asc("擦") Then        '检测 i 是否落在 B 首尾汉字拼音字母间
        拼音头字母  = "B"
    ElseIf i >= Asc("擦") And i < Asc("搭") Then        '检测 i 是否落在 C 首尾汉字拼音字母间
        拼音头字母  = "C"
    ElseIf i >= Asc("搭") And i < Asc("蛾") Then        '检测 i 是否落在 D 首尾汉字拼音字母间
        拼音头字母  = "D"
    ElseIf i >= Asc("蛾") And i < Asc("发") Then        '检测 i 是否落在 E 首尾汉字拼音字母间
        拼音头字母  = "E"
    ElseIf i >= Asc("发") And i < Asc("噶") Then        '检测 i 是否落在 F 首尾汉字拼音字母间
        拼音头字母  = "F"
    ElseIf i >= Asc("噶") And i < Asc("哈") Then        '检测 i 是否落在 G 首尾汉字拼音字母间
        拼音头字母  = "G"
```

```
    ElseIf i >= Asc("哈") And i < Asc("击") Then          '检测 i 是否落在 H 首尾汉字拼音字母间
        拼音头字母 = "H"
    ElseIf i >= Asc("击") And i < Asc("喀") Then          '检测 i 是否落在 J 首尾汉字拼音字母间
        拼音头字母 = "J"
    ElseIf i >= Asc("喀") And i < Asc("垃") Then          '检测 i 是否落在 K 首尾汉字拼音字母间
        拼音头字母 = "K"
    ElseIf i >= Asc("垃") And i < Asc("妈") Then          '检测 i 是否落在 L 首尾汉字拼音字母间
        拼音头字母 = "L"
    ElseIf i >= Asc("妈") And i < Asc("拿") Then          '检测 i 是否落在 M 首尾汉字拼音字母间
        拼音头字母 = "M"
    ElseIf i >= Asc("拿") And i < Asc("哦") Then          '检测 i 是否落在 N 首尾汉字拼音字母间
        拼音头字母 = "N"
    ElseIf i >= Asc("哦") And i < Asc("啪") Then          '检测 i 是否落在 O 首尾汉字拼音字母间
        拼音头字母 = "O"
    ElseIf i >= Asc("啪") And i < Asc("欺") Then          '检测 i 是否落在 P 首尾汉字拼音字母间
        拼音头字母 = "P"
    ElseIf i >= Asc("欺") And i < Asc("然") Then          '检测 i 是否落在 Q 首尾汉字拼音字母间
        拼音头字母 = "Q"
    ElseIf i >= Asc("然") And i < Asc("撒") Then          '检测 i 是否落在 R 首尾汉字拼音字母间
        拼音头字母 = "R"
    ElseIf i >= Asc("撒") And i < Asc("塌") Then          '检测 i 是否落在 S 首尾汉字拼音字母间
        拼音头字母 = "S"
    ElseIf i >= Asc("塌") And i < Asc("挖") Then          '检测 i 是否落在 T 首尾汉字拼音字母间
        拼音头字母 = "T"
    ElseIf i >= Asc("挖") And i < Asc("昔") Then          '检测 i 是否落在 W 首尾汉字拼音字母间
        拼音头字母 = "W"
    ElseIf i >= Asc("昔") And i < Asc("压") Then          '检测 i 是否落在 X 首尾汉字拼音字母间
        拼音头字母 = "X"
    ElseIf i >= Asc("压") And i < Asc("匝") Then          '检测 i 是否落在 Y 首尾汉字拼音字母间
        拼音头字母 = "Y"
    ElseIf i >= Asc("匝") And i <= Asc("座") Then         '检测 i 是否落在 Z 首尾汉字拼音字母间
        拼音头字母 = "Z"
    End If
End Function
```

4.3.7 修改按钮代码

【修改】按钮完成的任务是：对显示在当前窗体且已经被用户编辑好的客户记录做保存操作。单击该按钮后，首先会询问用户是否做出修改动作。单击【确定】按钮后，程序将会把当前窗体已编辑完成的客户记录所有项目依次写入客户表中，从而实现修改保存工作。该过程代码并不复杂，这里不再给出过程的流程图。以下是该过程的具体代码解释：

```
Private Sub 修改_Click()
    Dim i As Integer
    '准备修改选定的客户资料，首先询问用户是否做出修改操作
    If MsgBox("是否修改选定的客户资料？", vbQuestion + vbYesNo, "修改资料") = vbYes Then
    '将修改后的客户记录所有项目依次写入客户表中
        For i = 0 To UBound(myArray)                         '循环所有文本框和复合框
            Ws.Cells(intRowNow, i + 1) = Me.Controls(myArray(i)).Value      '保存客户信息
```

```
            Next
            Ws.Columns.AutoFit                          '客户表所有列自动适应，调整列宽
            '保存工作簿
            ThisWorkbook.Save
        End If
End Sub
```

4.3.8　删除按钮代码

【删除】按钮需要完成的工作比【修改】按钮要多。因为删除动作除了要完成删除数据、保存外，还需要确定下一条显示在窗体上的当前客户记录。并且在确定当前显示记录后，还需要修改 intRowNow 变量。该变量即当前显示客户记录在客户表中的行号，其作用就是随时保存当前客户记录在客户表中的行号，以便程序能够随时确认当前记录在客户表中的位置，而不需要再次进行查询定位。

过程首先询问用户是否做出删除操作。用户确认删除操作后，程序根据 intRowNow 变量记录的行号，在客户表中直接删除该行。随后程序根据客户表中剩余行数 intRowCount 以及 intRowNow 变量确认应该做出何种操作。

当 intRowCount 为 1 时，说明客户表中没有任何客户记录，此时将重置窗口中所有文本框和复合框。当 intRowCount 超过 1 且 intRowNow 大于 intRowCount 时，说明 intRowCount 行即为当前的显示行。当 intRowCount 超过 1 且 intRowNow 小于 intRowCount 时，说明 intRowNow 行即为当前的显示行。

完成所有客户资料信息显示后，程序需要重新设置各个浏览按钮的状态，以适应当前客户表中记录行情况。随后程序保存了删除客户记录后的工作表。其中设置按钮状态的过程请见本节后续内容。删除按钮单击事件过程的流程如图 4-23 所示。

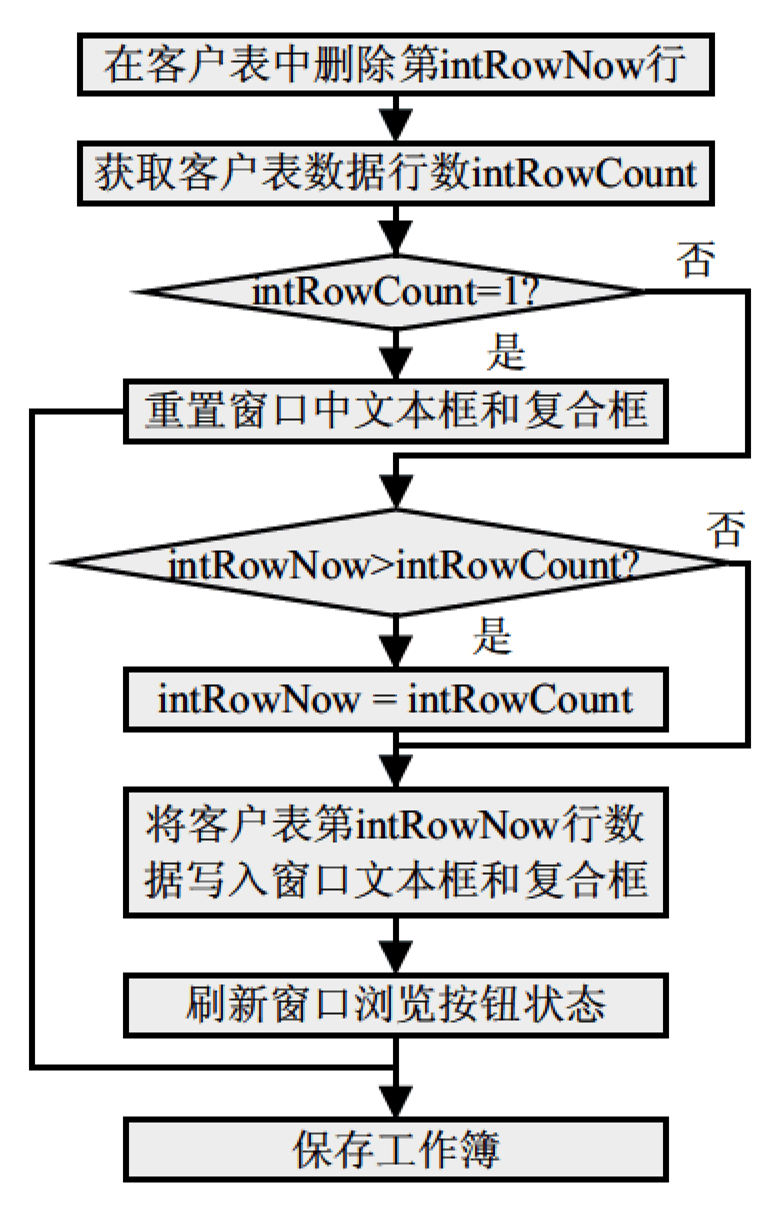


图 4-23　【删除】按钮单击事件过程流程图

以下是该按钮的详细代码说明。

```
Private Sub 删除_Click()
    Dim i As Integer, intRowCount As Integer
    '准备删除选定的客户资料
    If MsgBox("是否删除选定的客户资料？", vbQuestion + vbYesNo, "删除资料") = vbYes Then
        '在客户表中删除指定客户记录行
        ws.Rows(intRowNow).Delete Shift:=xlUp
        '获取删除记录后客户表中有数据行的行数
        intRowCount = Ws.Range("A" & Rows.Count).End(xlUp).Row
        '当总行数只有 1 行时，即只有标题行时，对窗体做重置操作，并做出提示
        '当总行数超过 1 行时，检测当前指向的行号是否超过最大行
        '当前行号超过最大行时，将当前行修改为最大行，即显示最后一条记录
        '当前行号小于最大行时，继续显示当前行
        If intRowCount = 1 Then
            '重置所有文本框和复合框控件
            For i = 1 To UBound(myArray) + 1            '循环所有文本框和复合框
                Me.Controls(myArray(i -1)).Value = ""   '置空控件显示数据
            Next
            '将焦点移到客户 ID 文本框
            客户 ID.SetFocus                            '鼠标焦点移动到客户 ID 文本框
            '设置地区文本框和国家文本框的默认值
            地区.ListIndex = 0
            国家.Value = "中国"
            MsgBox "客户记录表中已没有任何客户资料！", vbOKOnly + vbInformation
        Else
            If intRowNow > intRowCount Then            '检测 intRowNow 是否大于 intRowCount
                intRowNow = intRowCount                '修改 intRowNow 变量
            End If
            '将当前行的记录从客户表中依次写入窗体中
            For i = 0 To UBound(myArray)                '循环所有文本框和复合框
                Me.Controls(myArray(i)).Value = Ws.Cells(intRowNow, i + 1)   '显示客户数据到窗口
            Next
            提示.Caption = "当前记录在客户表中位于第" & intRowNow & "行"
            MsgBox "该条记录已经删除，当前显示的记录位于客户表的第" & intRowNow & "行",
vbOKOnly + vbInformation
        End If
        浏览按钮状态                                    '设置窗口中浏览按钮状态
        '保存工作簿
        ThisWorkbook.Save
    End If
End Sub
```

4.3.9 查看客户表按钮代码

【查看客户表】按钮可以显示和隐藏客户表。通过该按钮用户可以查看所有已经建立在系统中的所有客户资料。当第一次单击该按钮时，将会显示客户表，按钮的 Caption 属性会被修改为“隐藏客户表”。再次单击时，将会隐藏客户表，按钮的 Caption 属性被修改为“查看

客户表”。反复操作，依此类推。整个功能的实现使用了 Static 定义的一个局部静态变量。

该过程的代码并不复杂，以下不再列出该过程的流程图。其详细代码解释如下：

```
Private Sub 查看客户表_Click()
Static intCountClick As Integer
intCountClick = intCountClick + 1                    '累记该按钮被单击的次数
If intCountClick Mod 2 Then
    '单击奇数次时，执行以下代码
    查看工作表.Caption = "隐藏客户表"                '修改按钮 Caption 属性
    ws.Visible = xlSheetVisible                      '修改客户表 Visible 属性为可见
    ws.Activate                                      '激活客户表
Else
    ws.Visible = xlSheetVeryHidden                   '修改客户表 Visible 属性为不可见
    查看工作表.Caption = "查看客户表"                '修改按钮 Caption 属性
End If
End Sub
```

4.3.10　浏览按钮代码

在窗口中包含了 4 个用于浏览客户记录的按钮。这些按钮的功能与代码都不复杂，相互之间还有一定的相似性。因而本节将这几个按钮的代码放置在一起，以便读者阅读并加以比较。以下是各个按钮的功能描述：

- 【首条】按钮：单击该按钮时，窗口中显示的客户记录将会跳转到客户表的首条客户记录（即第二行记录）。在该按钮的单击事件过程中，还需要修改各个浏览按钮（首条、上一条、下一条和最后按钮）的可用状态以及提示标签。而修改各个浏览按钮的可用状态是通过“浏览按钮状态”过程完成的。该过程的具体代码介绍请见后续小节。
- 【上一条】按钮：单击该按钮时，将首先修改当前行的行数，以将当前行指向上一行，然后将该行记录所有项目显示到窗口中，并显示提示信息。
- 【下一条】按钮：该按钮单击事件的代码和上一条按钮的代码类似。唯一不同之处是：它把当前行的行号加 1，将所指向的行向后移动一行。
- 【最后】按钮：该按钮单击事件代码和首条按钮的代码类似。首先确定所要指向的当前行的行号，然后将当前行的所有项目显示到窗口中。

以下是各个按钮的详细代码解释：

```
Private Sub 首条_Click()
intRowNow = 2                                                    '修改当前行号为 2
浏览按钮状态                                                     '修改各浏览按钮的可用状态
'将当前记录显示到窗体中
For i = 0 To UBound(myArray)                                     '循环所有文本框和复合框
    Me.Controls(myArray(i)).Value = ws.Cells(intRowNow, i + 1)   '指定控件显示值
Next
提示.Caption = "当前记录在客户表中位于第" & intRowNow & "行"     '显示提示信息
End Sub
```

```
Private Sub 上一条_Click()
intRowNow = intRowNow -1                                    '修改当前行的行号
'将当前行的所有项目显示到窗口中
For i = 0 To UBound(myArray)                                '循环所有文本框和复合框
    Me.Controls(myArray(i)).Value = ws.Cells(intRowNow, i + 1)  '指定控件显示值
Next
浏览按钮状态                                                 '修改各个浏览按钮的可用状态
提示.Caption = "当前记录在客户表中位于第" & intRowNow & "行"    '显示提示信息
End Sub

Private Sub 下一条_Click()
intRowNow = intRowNow + 1                                   '修改当前行的行号
'将当前行的所有项目显示到窗口中
For i = 0 To UBound(myArray)                                '循环所有文本框和复合框
    Me.Controls(myArray(i)).Value = ws.Cells(intRowNow, i + 1)  '指定控件显示值
Next
浏览按钮状态                                                 '修改各个浏览按钮的可用状态
提示.Caption = "当前记录在客户表中位于第" & intRowNow & "行"    '显示提示信息
End Sub

Private Sub 最后_Click()
Dim intRowCount As Integer
intRowCount = ws.Range("A" & Rows.Count).End(xlUp).Row      '获取客户表中最后客户记录行的行号
intRowNow = intRowCount                                     '将当前行指向最后记录行
'将当前行的所有项目显示到窗口中
For i = 0 To UBound(myArray)                                '循环所有文本框和复合框
    Me.Controls(myArray(i)).Value = ws.Cells(intRowNow, i + 1)  '指定控件显示值
Next
浏览按钮状态                                                 '修改各个浏览按钮的可用状态
提示.Caption = "当前记录在客户表中位于第" & intRowNow & "行"    '显示提示信息
End Sub
```

4.3.11 浏览按钮状态过程代码设计

浏览按钮状态过程用于刷新窗口中浏览按钮的可用状态。该过程根据窗口中显示客户记录在客户表中的行号决定各个浏览按钮的可用状态。

当 intRowNow −2 值大于 0 时，说明当前行不在首条，【首条】按钮可用，否则【首条】按钮不可用。【上一条】按钮的可用状态和【首条】按钮的可用状态设置一样。当 intRowCount −intRowNow 值大于 0 时，说明当前显示记录行在客户表总行数前面。此时【下一条】按钮可用，否则【下一条】按钮不可用。【最后】按钮的可用状态和【下一条】按钮一样。

以下是该过程的详细代码解释：

```
Sub 浏览按钮状态()
Dim intRowCount As Integer
'获取客户表中最后客户记录行的行号
intRowCount = ws.Range("A" & Rows.Count).End(xlUp).Row
'当表达式 intRowNow -2 值大于 0 时，当前行不在首条，首条按钮可用
```

```
'当表达式 intRowNow -2 值小于等于 0 时，当前行在首条，首条按钮不可用
首条.Enabled = intRowNow -2
'上一条按钮的可用状态类似于首条按钮
上一条.Enabled = intRowNow -2
'当表达式 intRowCount -intRowNow 值大于 0 时，当前行在总行数前面，按钮可用
'当表达式 intRowCount -intRowNow 值小于等于 0 时，当前行在总行数后面，按钮不可用
下一条.Enabled = intRowCount -intRowNow
'最后按钮的可用状态与下一条类似
最后.Enabled = intRowCount -intRowNow
End Sub
```

4.4　客户资料查询导出窗体设计

客户资料查询导出模块完成客户资料的查询和导出工作。查询得到的客户资料会立即显示在窗口中的客户清单上。导出查询结果时，只需要单击窗体上的【输出报表】按钮即可。用户在查询客户资料时，可以采用 3 种方式：

- 通过在筛选列表框中选择筛选条件查询客户资料。窗口中包含了 3 个列表框，它们分别对应建立客户资料时的国家、地区和城市 3 个项目。当需要将客户按照所在区域位置划分时，这种查询比较方便。
- 按客户名称查询客户资料。用户可以采用 3 种客户名称输入方式：按全名、按缩写和按拼音。选择查询方式后，在【输入条件值】文本框中输入查询值后单击【开始查询】按钮即可。
- 按其他项目查询客户资料。按其他项目查询需要设置 3 项查询参数。在【选择项目】下拉列表框中输入项目名，这些项目名对应客户资料建立时的所有项目。选择匹配符支持等于和 Like（类似）两种匹配方式，然后在【输入条件值】文本框中输入查询值即可。

客户信息管理窗体的界面如图 4-24 所示。



图 4-24　客户资料查询导出窗体界面

4.4.1 窗体界面设计

客户资料查询导出窗体界面的控件按照其功能可以划分为 5 块，它们分别是按区域筛选列表框、按客户名称查询、按其他项目查询、功能按钮区和查询结果清单。

1．按区域筛选列表框

该区域仅包含 3 个列表框，分别指向建立客户资料时的 3 个项目：国家、地区和城市。在窗体初始化时，【选择国家】列表框会获取初始值。在【选择国家】列表框中做出选择后，【选择地区】列表框会获取当前国家下所有的地区值。在【选择地区】列表框中做出选择后，【选择城市】列表框会获取当前国际、地区下的城市值。在以上做出选择的同时，所有符合当前区域筛选条件的客户资料会被立即显示在下面的客户清单上。

3 个列表框控件的 ListStyle 属性在窗体初始化时被设置为 fmListStyleOption。列表框中各个选项的左方将显示选项按钮。

2．按客户名称查询

该区域包括 3 个单选按钮、1 个文本框。3 个单选按钮设置按客户名称查询的查询方式。在文本框中输入对应该查询方式下的查询条件字符串。

3．按其他项目查询

该区域包括 2 个列表框、1 个文本框。两个列表框共同确定查询的查询方式。【选择项目】下拉列表框决定筛选的项目对象。【选择匹配符】下拉列表框决定筛选的运算方式。设置匹配符为等于时，输入条件需要输入完整条件；设置为 Like 时可以模糊查询。

4．查询结果清单

该区域仅包括一个 ListView 控件，该控件在 VBE 的窗体设计的默认工具箱中不存在。调出该控件的步骤如下：

（1）首先依次选择 Excel 2007 VBE 中的【工具】|【引用】命令。在随后显示的【引用】对话框中选择【Microsoft Windows Common Control 6.0】选项，然后单击【确定】按钮即可，如图 4-25 所示。

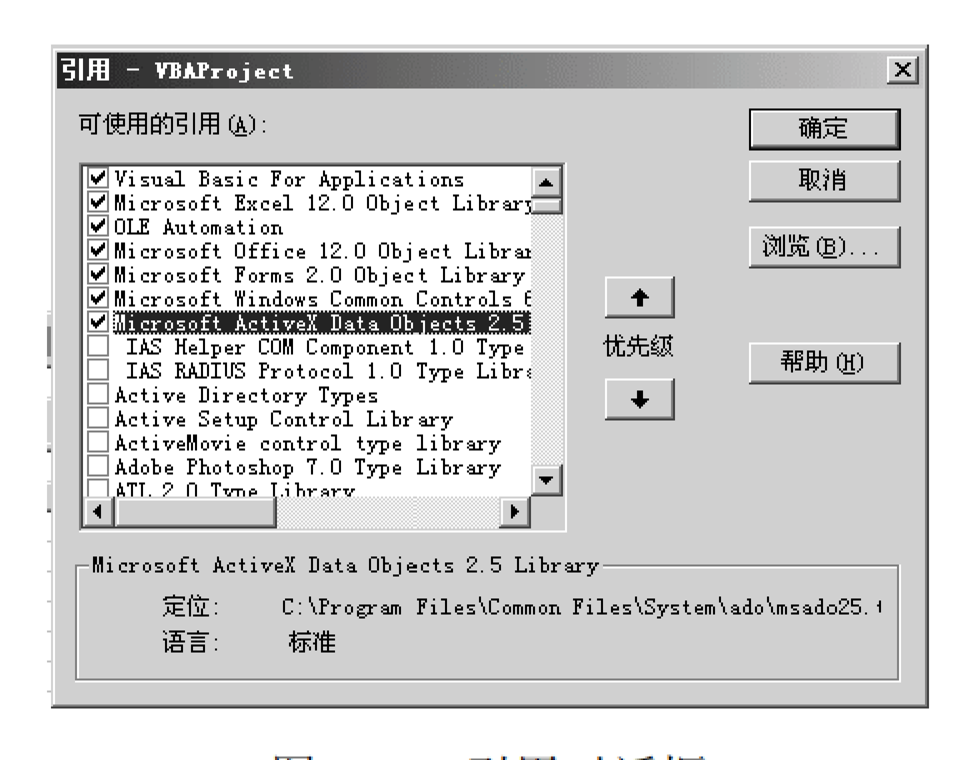


图 4-25　引用对话框

（2）在工具箱的空白区域右击，在弹出的快捷菜单中选择【附加控件】命令，打开【附加控件】对话框，如图 4-26 所示。在【附加控件】对话框中选择【Micorsoft ListView Control 6.0 (SP6)】项目，然后点击【确定】按钮，如图 4-27 所示。

（3）工具箱中将会多出一个按钮，此时工具箱的实际效果如图 4-26 所示。单击该按钮后，再在窗体的设计器中单击鼠标左键即可在窗体中插入 ListView 控件。

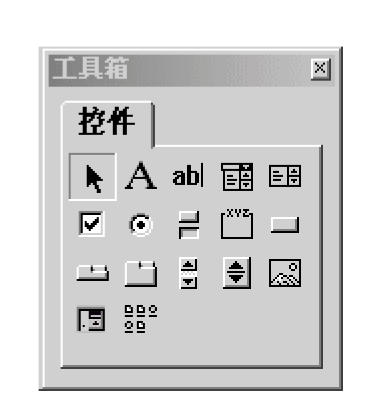


图 4-26　工具箱

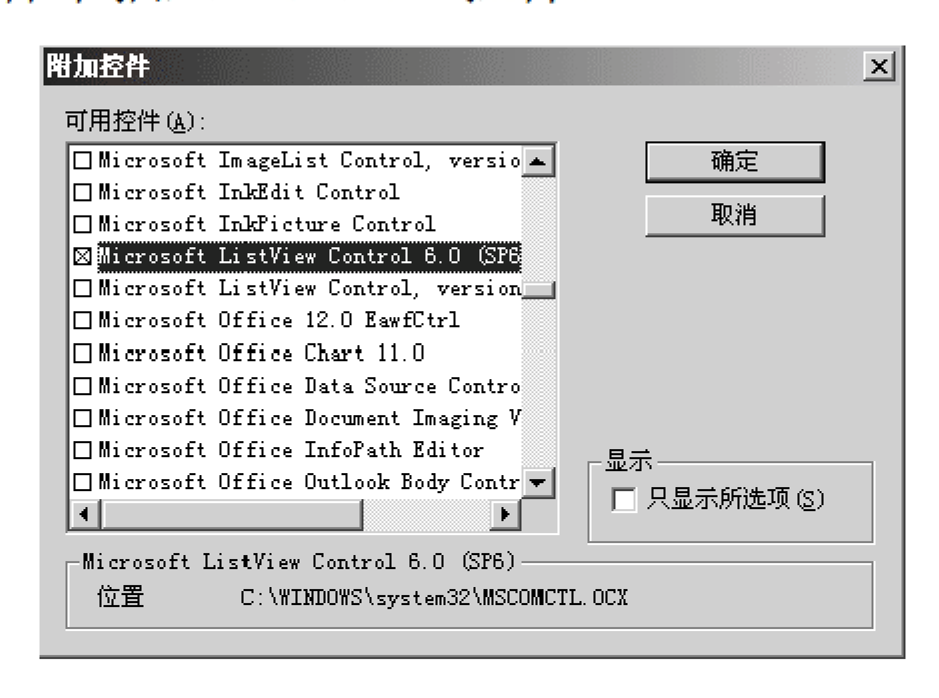


图 4-27　【附加控件】对话框

4.4.2　窗体初始化代码

客户资料查询导出窗口的初始化工作大体包括：初始化公共变量、初始化控件初始状态。需要初始化的公共变量包括客户表工作表对象和客户资料项目数组 myArray。需要初始化的控件比较多，下面分类讲解。

按区域查询中的 3 个列表框需要初始化显示模式和提示文本。ListView 控件需要初始化标题、显示类型、整行选择、网格线及排序属性。【按其他项目查询】选项组中的列表框需要初始化项目值与初始值，其文本框需要初始化提示文本。该过程中没有复杂的结构控制语句，程序的执行顺序没有中途发生任何跳转。其流程如图 4-28 所示。



图 4-28　窗口初始化流程图

以下是该初始化过程的详细代码解释：

```
Private Sub UserForm_Initialize()
    Dim SQL As String, i As Integer
    '设置按区域查询中的 3 个列表框 ListStyle 属性
    ListBox1.ListStyle = fmListStyleOption                '国家列表框外观模式
    ListBox2.ListStyle = fmListStyleOption                '地区列表框外观模式
    ListBox3.ListStyle = fmListStyleOption                '城市列表框外观模式
    '设置按区域查询中的 3 个列表框的提示文本
    ListBox1.ControlTipText = "单击或双击某个国家名称，显示该国家的全部客户信息"
    ListBox2.ControlTipText = "单击某个地区名称，显示该地区的全部客户信息"
    ListBox3.ControlTipText = "单击某个城市名称，显示该城市的全部客户信息"
    '初始化工作表对象和客户资料项目数组 myArray
    Set ws = Worksheets("客户表")                          '设定工作表对象
```

```
myArray = Array("客户 ID", "客户名称", "名称缩写", "负责人", "地址", _
    "邮政编码", "城市", "地区", "国家", "电话", "传真", "Email", "网址", _
    "经营范围", "客户级别", "信用等级", "联系人", "联系人电话", _
    "联系人 Email", "备注")
'设置 ListView1 控件的标题、显示类型、整行选择、网格线及排序属性
With ListView1
    .ColumnHeaders.Clear                          '清除标题
    .ListItems.Clear                              '清除 ListView1 控件的项目
    .View = lvwReport                             '设置显示类型
    .FullRowSelect = True                         '整行选择
    .Gridlines = True                             '显示网格线
    .Sorted = True                                '对 ListView1 中的项目排序
    '初始化 ListView1 的标题
    For i = 0 To UBound(myArray)                  '循环 myArray 数组中所有元素
        .ColumnHeaders.Add , , myArray(i)         '为复合框添加标题
    Next
End With
'建立与工作簿的连接
Set cnn = New ADODB.Connection                    '新建数据库链接
cnn.Open "Provider=microsoft.jet.oledb.4.0;" _
    & "Extended Properties=Excel 8.0;" _
    & "Data Source=" & ThisWorkbook.FullName      '指定数据库链接字符串
'查询所有国家名称，并设置给列表框 ListBox1
myCountry                                         '调用 myCountry 过程
'将“按客户名称查询”和“按全名”选项按钮设置为默认按钮
按全名.Value = True                               '选中按全名单选按钮
按客户名称查询.Value = True                       '选中按客户名称查询单选按钮
'查询各个项目的不重复值，并设置给“查询项目”复合框
SQL = "select * from [客户表$]"                   '设置查询字符串
Set rs = New ADODB.Recordset                      '获得新的查询记录集对象
rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic  '生成查询记录集
'设置在按其他条件插叙中的文本框控件的提示文本
条件值.ControlTipText = "如果不输入任何条件值，并且匹配符选择 LIKE，那么就查询全部记录。"
'初始化按其他项目查询中查询选择项目复合框
With 查询项目
    .Clear                                        '清空查询项目中所有项目
    For i = 0 To rs.Fields.Count -1               '循环记录集中所有记录
        .AddItem rs.Fields(i).Name                '为查询项目添加项目
    Next
    .ListIndex = 0                                '设置该复合框的初始显示值
End With
'为“匹配符”复合框设置项目
With 匹配符
    .Clear                                        '清空匹配符复合框所有项目
    .AddItem "="                                  '为匹配符复合框添加第一个项目
    .AddItem "like"                               '为匹配符复合框添加第二个项目
    .ListIndex = 0                                '设置匹配符复合框默认显示值
End With
'查询所有国家的客户信息，并将有挂信息显示在窗体上
```

```
    myCountry                                      '重新显示客户信息
End Sub
```

4.4.3　myCountry 与 myList 过程代码设计

在窗体初始化过程代码中调用了一个自定义公共过程 myCountry。该公共过程首先获取客户表国家字段中不重复项的一个记录集对象，然后调用 myList 公共过程。myList 公共过程将不重复的国家字段记录集中的记录写入到当前复合框控件中。myList 过程接受 myStr 和 myListBox 两个参数。

以下是 myCountry 和 myList 公共过程的详细代码解释：

```
Public Sub myCountry()
    Dim SQL As String
    SQL = "select distinct 国家 from [客户表$]"        '指定查询记录字符串
    Set rs = New ADODB.Recordset                       '建立新数据库记录集对象
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic  '打开数据库查询记录集
    myList("国家", Me.ListBox1)               '将查询获取的所有不重复国家写入国家列表框中
End Sub

Public Sub myList(myStr As String, myListBox As MSForms.ListBox)
Dim i As Integer
On Error Resume Next
With myListBox
    .Clear                                  '清空列表框中所有项目
    For i = 1 To rs.RecordCount             '循环记录集中所有记录
        .AddItem rs.Fields(0).Value         '为列表框添加新项目
        rs.MoveNext                         '移动记录集指针到下一条记录
    Next
End With
End Sub
```

4.4.4　按区域筛选客户代码设计

在窗口的左上方包含了 3 个列表框，用户可以通过这 3 个列表框对客户记录按区域进行查询。这 3 个列表框分别用于选择国家、地区、城市，选择的顺序是从左到右。3 个列表框的事件代码主要是列表的单击事件。以下分别介绍这 3 个列表框的单击事件的功能：

- ❑ 国家列表框单击事件：当用户在【选择国家】列表框中选择了某个国家时，在【选择地区】列表框中将刷新出当前国家中所有地区，同时下方的客户清单控件将会显示当前选择国家的所有记录。
- ❑ 地区列表框单击事件：当用户在【选择地区】列表框中选择了某个地区时，在【选择城市】列表框中将刷新出当前国家地区下所有城市，同时下方的客户清单控件将会显示当前选择国家与地区的所有记录。
- ❑ 城市列表框单击事件：当用户在【选择城市】列表框中选择某个城市时，此时不需

要再改变其他列表框的数据，只需要将下方的客户清单控件中显示记录刷新为当前国家、地区和城市的记录即可。

以下是这 3 个列表框的详细事件代码解释：

```
Private Sub ListBox1_Click()
Dim SQL As String
'重置选项按钮、文本框的值
按全名.Value = True                                          '重置按全名选项按钮的值
客户名.Value = ""                                            '置空客户名文本框
条件值.Value = ""                                            '置空条件值文本框
'为地区列表框设置项目
SQL = "select distinct 地区 from [客户表$] where 国家='" & ListBox1.Value & "'"    '设置查询字符串
Set rs = New ADODB.Recordset                                 '新建数据库记录集
rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic             '打开数据库查询记录集
myList "地区", Me.ListBox2                                   '显示地区列表框中的项目
'清除城市列表框的项目
Me.ListBox3.Clear
'查询指定国家的所有客户信息
SQL = "select * from [客户表$] where 国家='" & ListBox1.Value & "'"               '设置查询字符串
Set rs = New ADODB.Recordset                                 '新建数据库记录集
rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic             '打开数据库查询记录集
'调用子程序，为 ListView1 控件输入数据
myListView                                                   '将查询结果显示在客户清单控件中
End Sub

Private Sub ListBox1_DblClick(ByVal Cancel As MSForms.ReturnBoolean)
ListBox1_Click                                               '国家列表框双击时执行单击事件过程
End Sub

Private Sub ListBox2_Click()
Dim SQL As String
'重置选项按钮、文本框的值
按全名.Value = True
客户名.Value = ""
条件值.Value = ""
'为城市列表框设置项目
SQL = "select distinct 城市 from [客户表$]" _
        & " where 国家='" & ListBox1.Value & "'" _
        & " and 地区='" & ListBox2.Value & "'"               '设置查询字符串
Set rs = New ADODB.Recordset                                 '新建数据库记录集
rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic             '打开数据库查询记录集
myList "城市", Me.ListBox3                                   '显示地区列表框中的项目
'查询指定国家和地区的所有客户信息
SQL = "select * from [客户表$] " _
        & " where 国家='" & ListBox1.Value & "'" _
        & " and 地区='" & ListBox2.Value & "'"               '设置查询字符串
Set rs = New ADODB.Recordset                                 '新建数据库记录集
rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic             '打开数据库查询记录集
'调用子程序，为 ListView1 控件输入数据
```

```
myListView                                                '将查询结果显示在客户清单控件中
End Sub

Private Sub ListBox3_Click()
Dim SQL As String
'重置选项按钮、文本框的值
按全名.Value = True
客户名.Value = ""
条件值.Value = ""
'查询指定国家、地区和城市的所有客户信息
SQL = "select * from [客户表$] " _
        & " where 国家='" & ListBox1.Value & "'" _
        & " and 地区='" & ListBox2.Value & "'" _
        & " and 城市='" & ListBox3.Value & "'"          '设置查询字符串
Set rs = New ADODB.Recordset                              '新建数据库记录集
rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic          '打开数据库查询记录集
'调用子程序，为 ListView1 控件输入数据
myListView                                                '将查询结果显示在客户清单控件中
End Sub
```

4.4.5　myListView 过程代码设计

在前面的按区域筛选客户中，3 个列表框的单击事件代码最后都调用了一个过程 myListView。该过程用于将查询结果显示到客户清单控件中。该过程的代码比较多，因而单独列了一小节。

程序根据用户是否选中【按拼音查询】单选按钮，分别执行分支代码。该分支主要是由于按拼音查询时，需要检测记录中客户名称的拼音是否与输入的拼音一致造成的。执行完以上操作以后，程序为客户清单控件的各个列设置宽度。

以下是该过程的详细代码解释：

```
Public Sub myListView()
    On Error Resume Next
    Dim i As Integer, j As Long
    '将查询结果显示在 ListView1 控件中
    If 按拼音.Value = False Then
        With ListView1
            .ListItems.Clear                                   '清除控件所有项目
            For i = 1 To rs.RecordCount                        '循环记录集所有记录
                .ListItems.Add , , rs.Fields(0).Value          '为控件添加新项目
                For j = 1 To rs.Fields.Count -1                '循环记录所有字段
                    If IsNull(rs.Fields(j).Value) Then         '检测记录字段是否为空
                        .ListItems(i).SubItems(j) = ""         '为空时，子项显示空字符串
                    Else
                        .ListItems(i).SubItems(j) = rs.Fields(j).Value   '设定子项的数据
                    End If
                Next
```

```
                rs.MoveNext                              '将记录集指针移动到下一条记录
            Next
        End With
        rs.MoveFirst                                     '将记录集指针移动到首条记录
    Else
        With ListView1
            .ListItems.Clear                             '清除控件所有项目
            For i = 1 To rs.RecordCount                  '循环记录集所有记录
                '检测当前记录的客户名拼音是否与输入拼音一致
                If 检查拼音(rs.Fields("客户名称"), intRow) Then
                    .ListItems.Add , , rs.Fields(0).Value    '为控件添加新项目
                    For j = 1 To rs.Fields.Count -1          '循环记录所有字段
                        If IsNull(rs.Fields(j).Value) Then   '检测记录字段是否为空
                            .ListItems(i).SubItems(j) = ""   '为空时，子项显示为空字符串
                        Else
                            .ListItems(i).SubItems(j) = rs.Fields(j).Value    '设定子项的数据
                        End If
                    Next
                End If
                rs.MoveNext                              '将记录集指针移动到下一条记录
            Next
        End With
        rs.MoveFirst                                     '将记录集指针移动到首条记录
    End If
    '自动设置 ListView1 控件各列的宽度
    For i = 1 To ListView1.ColumnHeaders.Count                      '循环控件所有标题列
        ListView1.ColumnHeaders(i).Width = ws.Cells(1, i).Width * 0.9    '设置标题列的宽度
    Next
End Sub
```

4.4.6 选项按钮、文本框和复合框代码设计

本小节包含讲述的控件比较多，但是各个控件的代码都十分简单。这些控件的事件代码基本上都是用于设置控件的某个属性而用的，其中部分事件代码调用了 ListPriValue 过程。该过程用于设置 3 个按区域筛选客户的列表框以及客户清单控件的显示项目。以下是这些控件的事件代码的详细解释：

```
Private Sub 按客户名称查询_Click()
    查询项目.Value = ""                    '置空查询项目复合框
    匹配符.Value = ""                      '置空匹配符复合框
    条件值.Value = ""                      '置空条件值文本框
    ListPriValue                           '设置 3 个列表框控件和客户清单控件
End Sub

Private Sub 按其他项目查询_Click()
    按全名.Value = False                   '取消按全名单选按钮选中状态
    按缩写.Value = False                   '取消按缩写单选按钮选中状态
```

```
    按拼音.Value = False                                '取消按拼音单选按钮选中状态
    客户名.Value = ""                                   '置空客户名文本框
End Sub

Private Sub 按全名_Enter()
    按客户名称查询.Value = True                         '选中按客户名称查询单选按钮
     ListPriValue                                       '设置 3 个列表框控件和客户清单控件
End Sub

Private Sub 按缩写_Enter()
    按客户名称查询.Value = True                         '选中按客户名称查询单选按钮
     ListPriValue                                       '设置 3 个列表框控件和客户清单控件
End Sub

Private Sub 按拼音_Enter()
    按客户名称查询.Value = True                         '选中按客户名称查询单选按钮
     ListPriValue                                       '设置 3 个列表框控件和客户清单控件
End Sub

Private Sub 客户名_Enter()
    按客户名称查询.Value = True                         '选中按客户名称查询单选按钮
     ListPriValue                                       '设置 3 个列表框控件和客户清单控件
End Sub

Private Sub 查询项目_Enter()
    按其他项目查询.Value = True                         '选中按其他项目查询单选按钮
     ListPriValue                                       '设置 3 个列表框控件和客户清单控件
End Sub

Private Sub 匹配符_Enter()
    按其他项目查询.Value = True                         '选中按其他项目查询单选按钮
     ListPriValue                                       '设置 3 个列表框控件和客户清单控件
End Sub

Private Sub 条件值_Enter()
    按其他项目查询.Value = True                         '选中按其他项目查询单选按钮
     ListPriValue                                       '设置 3 个列表框控件和客户清单控件
End Sub

Public Sub ListPriValue()
    ListBox1.ListIndex = -1                             '不选中国家列表框中的任何项目
    ListBox2.ListIndex = -1                             '不选中地区列表框中的任何项目
    ListBox3.ListIndex = -1                             '不选中城市列表框中的任何项目
    ListBox2.Clear                                      '清空地区列表框所有项目
    ListBox3.Clear                                      '清空城市列表框所有项目
    ListView1.ListItems.Clear                           '清空客户清单控件所有项目
End Sub
```

4.4.7 开始查询按钮单击事件代码设计

【开始查询】按钮使用当前用户输入的查询条件完成查询工作，然后将查询结果显示在窗口的客户清单控件中。

单击【开始查询】按钮后，程序将根据用户的查询设置条件，定义数据库查询字符串。从窗口中可以看出按客户名称查询和按其他项目查询的查询字符串显然是不同的。而这两个情况下还有其他各项设置，当选择不同设置时，查询字符串也不一样。

获取了相应的查询字符串之后，程序将打开该查询记录集。该记录集中没有客户名称拼音字头字段，因此当用户选择了按拼音查询时，需要检测客户名称拼音字头是否与用户输入一致。这里程序通过一个 If 结构分情况处理。最后，程序将查询到的符合条件的记录显示在客户清单控件中。

该按钮的单击事件代码解释如下：

```
Private Sub 开始查询_Click()
    Dim SQL As String
    SQL = "select * from [客户表$]"                          '设置查询字符串
    If 按客户名称查询.Value = True Then                       '检测按客户名称查询单选按钮是否选中
        If 按全名.Value = True Then                           '检测按全名单选按钮是否选中
            SQL = SQL & " where 客户名称='" & 客户名.Value & "'"    '设置按全名时的查询字符串
        ElseIf 按缩写.Value = True Then                       '检测按缩写单选按钮是否选中
            SQL = SQL & " where 名称缩写='" & 客户名.Value & "'"    '设置按缩写时的查询字符串
        ElseIf 按拼音.Value = True Then                       '检测按拼音单选按钮是否选中
            SQL = SQL                                         '设置按拼音时的查询字符串
        End If
    ElseIf 按其他项目查询.Value = True Then                   '检测按其他项目查询单选按钮是否选中
        If 匹配符.Value = "=" Then                            '检测匹配符是否选择等于号
            '设置匹配符为等于号时的查询字符串
            SQL = SQL & " where " & 查询项目.Value & "='" & 条件值.Value & "'"
        Else
            '设置匹配符为 Like 时的查询字符串
            SQL = SQL & " where " & 查询项目.Value & " like '%" & 条件值.Value & "%'"
        End If
    End If
    Set rs = New ADODB.Recordset                              '新建数据库记录集
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic          '从数据库中获取记录集
    If 按拼音.Value = False Then                              '检测按拼音单选按钮是否选中
        If rs.EOF And rs.BOF Then                             '检测记录集是否为空
            MsgBox "没有查询到符合条件的记录！", vbInformation, "查询结果"
        Else
            myListView                                        '刷新客户清单控件项目
        End If
    Else
        If 检查拼音(客户名.Value) = True Then                 '检测客户名称拼音字头
            myListView                                        '刷新客户清单控件项目
        End If
```

```
    End If
End Sub
```

4.4.8　输出报表过程代码设计

输出报表过程用于将当前查询到的结果保存在一个新 Excel 文件中。该程序首先新建了一个工作簿，然后循环记录集的字段，将字段名作为新工作表的标题行，接着从该记录集直接将数据复制到新工作表中，最后将工作表的所有列自动对齐。该过程的代码比较简单，流程并不复杂，以下不再列出该过程的流程图。

以下是该过程的代码解释：

```
Private Sub 输出报表_Click()
    Dim i As Integer
    Dim wb As Workbook
    Set wb = Workbooks.Add                              '新建工作簿
    With wb.ActiveSheet
        For i = 1 To rs.Fields.Count                    '循环记录集所有字段
            .Cells(1, i) = rs.Fields(i -1).Name         '将字段名写入工作表的标题
        Next
        .Range("A2").CopyFromRecordset rs               '从记录集中直接复制数据到 A2 单元格
        .Columns.AutoFit                                '工作表自动对齐
    End With
End Sub
```

4.5　系统测试

本系统中所有功能都是通过首页的两个按钮实现的，这两个按钮分别打开相应的窗口，用户在这两个窗口中分别完成相应的工作。下面以具体的操作流程来演示系统的操作方式。基于系统的两个窗口，以下将分两个小节分别介绍两窗口的操作。

4.5.1　客户资料管理窗口测试

该窗口中需要测试的是几个按钮的功能是否准确实现。其中新增、修改、删除以及逐条浏览按钮的代码十分简单。此处不再加以具体测试，用户可以自己打开工作簿后，测试这些按钮的功能。本小节讲述的主要是【查找】和【查看客户表】按钮的功能。查找按钮的代码稍显复杂，而查看客户表按钮将激活客户表。以下是这两个按钮的测试过程。

（1）在工作簿的首页单击【客户资料管理】标签按钮，打开【客户资料管理】窗口（如图 4-29 所示），其中被画圈的按钮为需要测试的两个按钮。窗口当前显示的记录为客户表中第一条客户记录，因而此时【首条】与【上一条】按钮都为不可用。

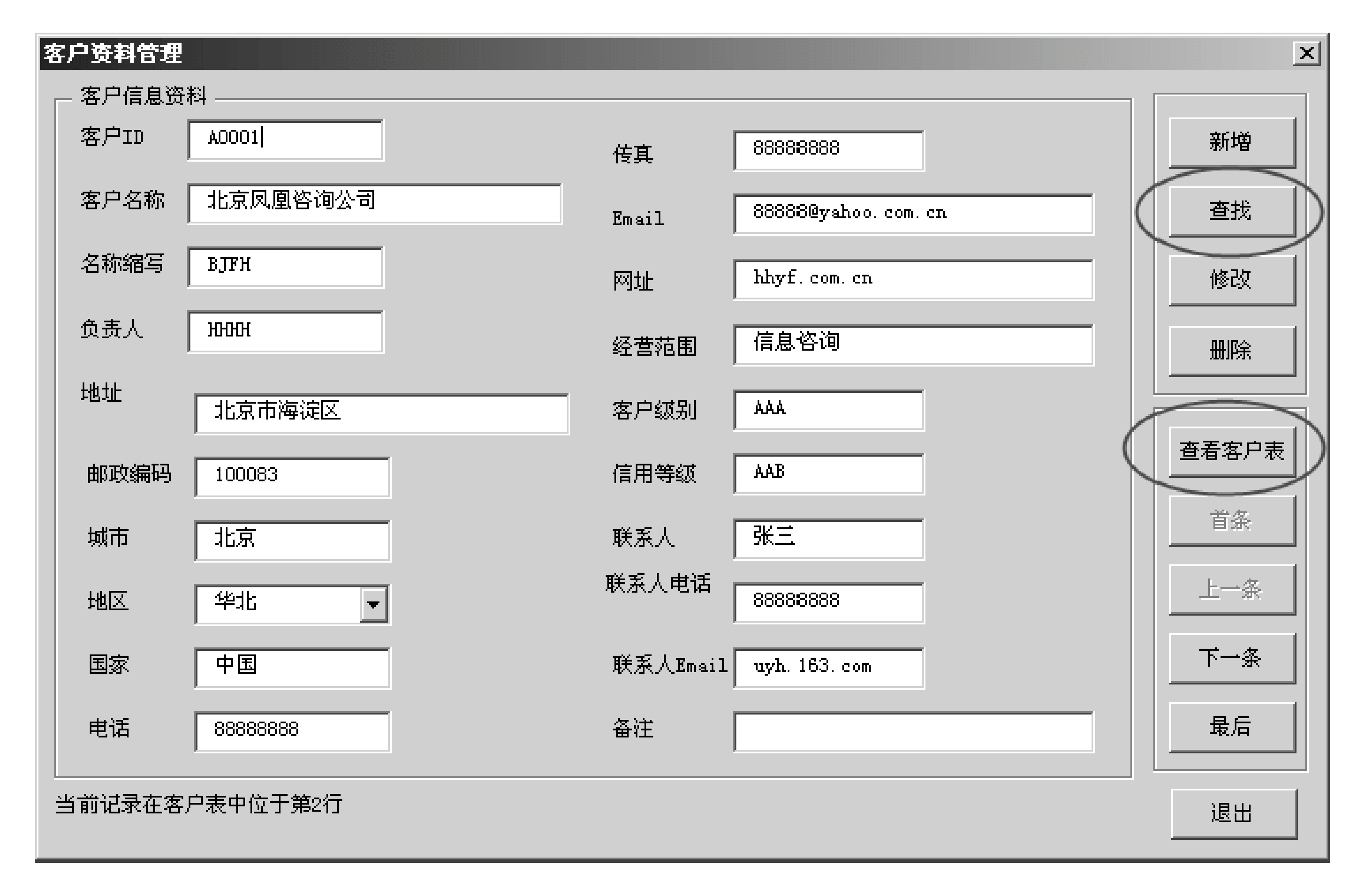


图 4-29　客户资源管理窗口测试

（2）单击【查找】按钮，将打开一输入框（如图 4-30 所示），这里输入“shdfgs”，单击【确定】按钮。

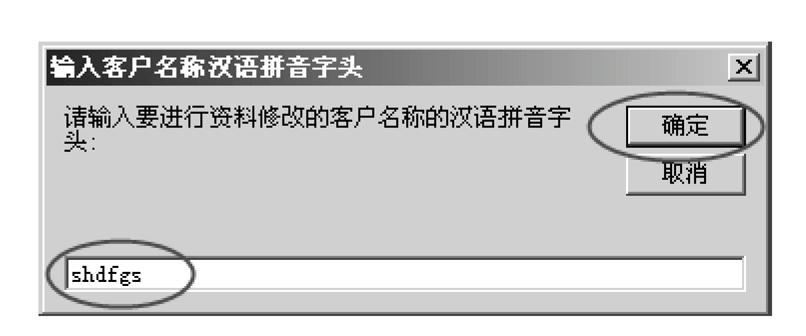


图 4-30　客户名称拼音字头输入框

（3）程序将在客户表中查找客户名称拼音字头为“shdfgs”的记录，在客户表中存在一条“上海东方公司”的记录与之对应。此时，窗口将定位到该记录并将其显示到窗口中，查找结果如图 4-31 所示。从图中提示中可以知道该记录位于客户表的第四行。此时，【首条】和【上一条】按钮的可用状态都被恢复。客户表一共 5 行，所以【下一条】和【最后】按钮都可用。

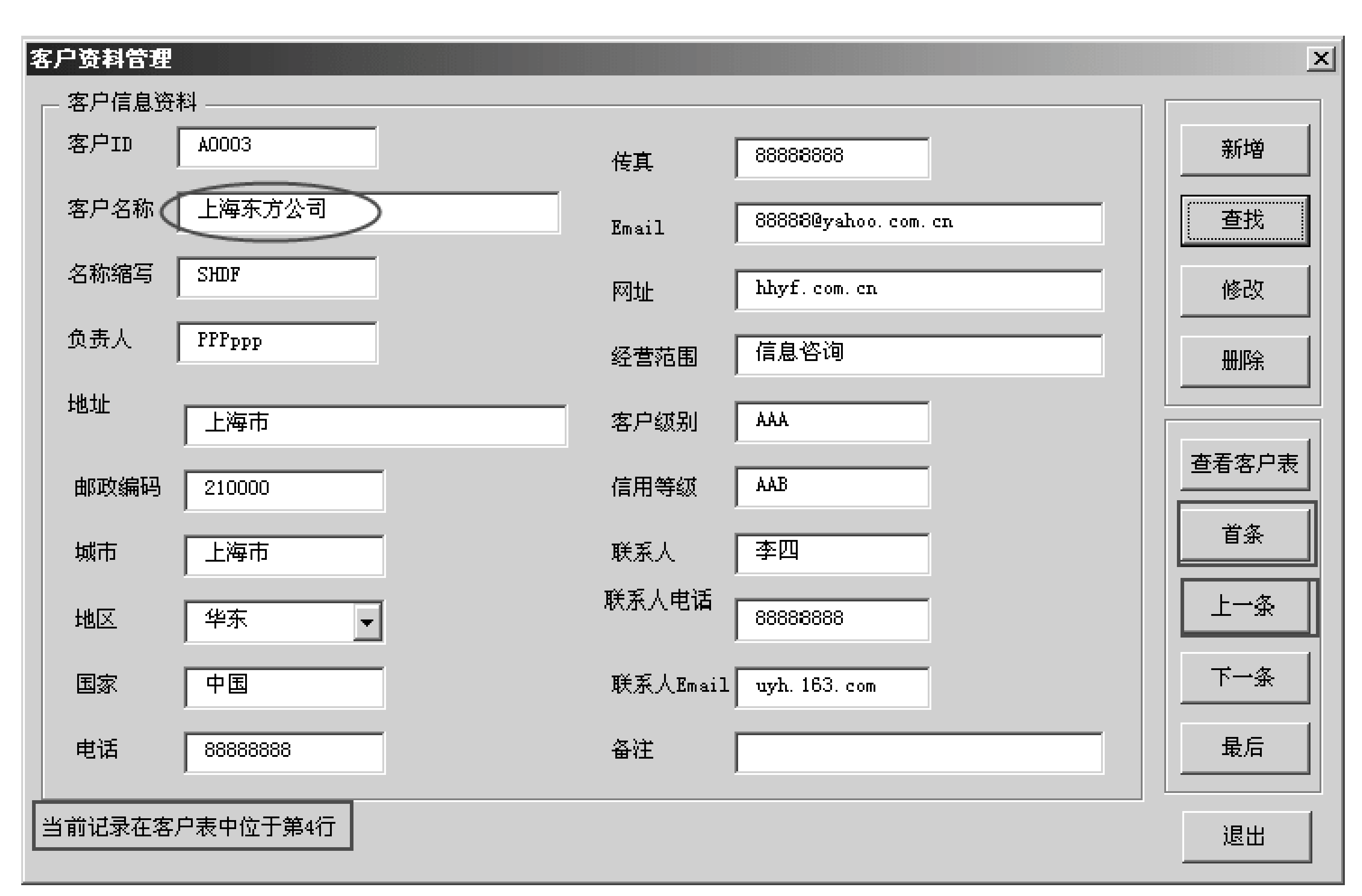


图 4-31　查找结果显示

（4）在窗口中单击【查看客户表】按钮，此时客户表将会被显示出来（如图 4-32 所示）。用户可以在表中查看所有已经建立资料的客户记录。

客户管理系统.xlsm

	A	B	C	D	E	F	G	H	I	J	K	L
1	客户ID	客户名称	名称缩写	负责人	地址	邮政编码	城市	地区	国家	电话	传真	Email
2	A0001	北京凤凰咨询公司	BJFH	HHHH	北京市海淀区	100083	北京	华北	中国	88888888	88888888	88888@yahoo.c
3	A0002	河北维维公司	HBWW	kkkk	河北省石家庄	055155	石家庄	华北	中国	88888888	88888888	88888@yahoo.c
4	A0003	上海东方公司	SHDF	PPPppp	上海市	210000	上海市	华东	中国	88888888	88888888	88888@yahoo.c
5	A0004	浙江北方公司	ZJBF	TTT	浙江省杭州市	310000	杭州市	华东	中国	88888888	88888888	88888@yahoo.c
6												

首页　客户表

图 4-32　查看客户表所有记录

4.5.2　客户资料查询导出窗口测试

在【客户资料查询导出】窗口中，一共包含了 3 个查询方式，分别是按区域查询、按客户名称查询和按其他项目查询。这里只介绍按区域查询操作过程。其测试操作过程如下：

（1）在首页单击【客户资料查询导出】标签，打开【客户资料查询导出】窗口（如图 4-33 所示），图中已经在【选择国家】列表框中选中了【选择中国】单选按钮，此时可以看到客户清单中已经将所有国家字段为“中国”的记录显示出来，而【选择地区】列表框中也显示了所有地区项目。



图 4-33　客户资料查询导出测试

（2）在【选择地区】列表框中选中【华东】单选按钮，此时查询的记录是所有国家为“中国”、地区为“华东”的记录，其查询结果如图 4-34 所示。从图中可以看出客户清单已经显示了所有查询结果，并且【选择城市】列表框中也显示了所有的城市。

（3）在【选择城市】列表框中选择【杭州市】，此时查询记录就是所有国家为“中国”、地区为“华东”、城市为“杭州市”的记录，查询结果如图 4-35 所示。

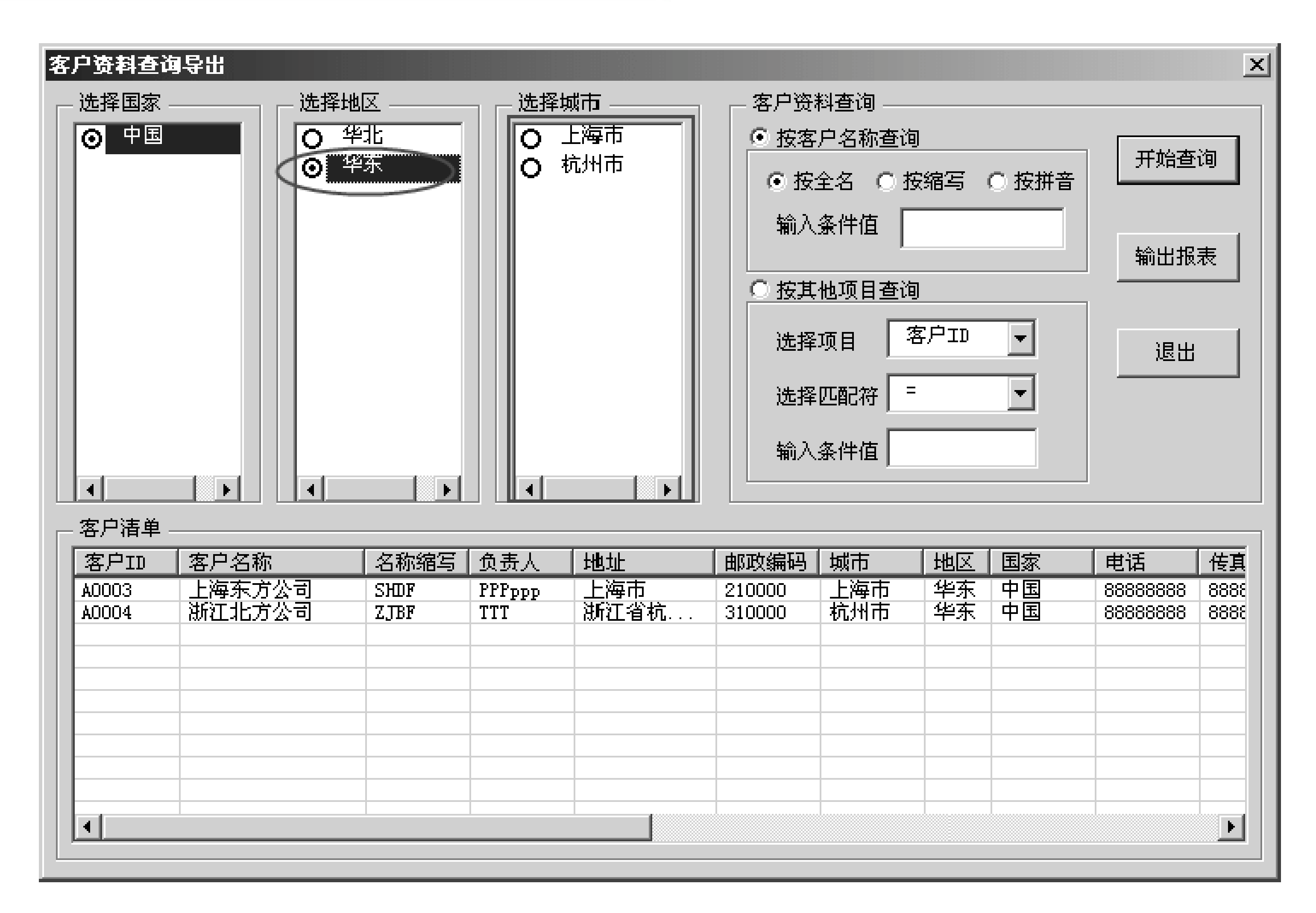


图 4-34 选择国家与地区查询结果



图 4-35 选择国家、地区和城市查询结果

第 5 章　学生成绩管理系统

学生成绩管理是学校教学管理的一个重要组成部分。根据学校的具体需求，建立一套完善的学生成绩管理系统，实现成绩管理的数字化、信息化，可以大幅度减轻教务工作人员的工作强度，提高工作效率。

本实例主要针对中小型学校的成绩管理工作设计，借助了 Excel 的常用功能及其便利的输入环境，提高了实例的可操作性和实用性。

5.1　系 统 概 述

本例大量使用了 Excel 2007 本身自带的功能，如数据有效性、自动筛选、冻结窗口等。实例以工作表来表现操作界面，包含的代码简单易懂，整体架构清晰，可以轻松辅助使用者完成学生成绩建立、查询与编辑工作。

5.1.1　设计思路

本实例基于学校成绩管理的需求，需要完成输入、分析和查询功能。将系统整体架构划分为 3 大功能模块，即基本资料建立模块、成绩输入与分析模块和查询模块。系统详细结构如图 5-1 所示。

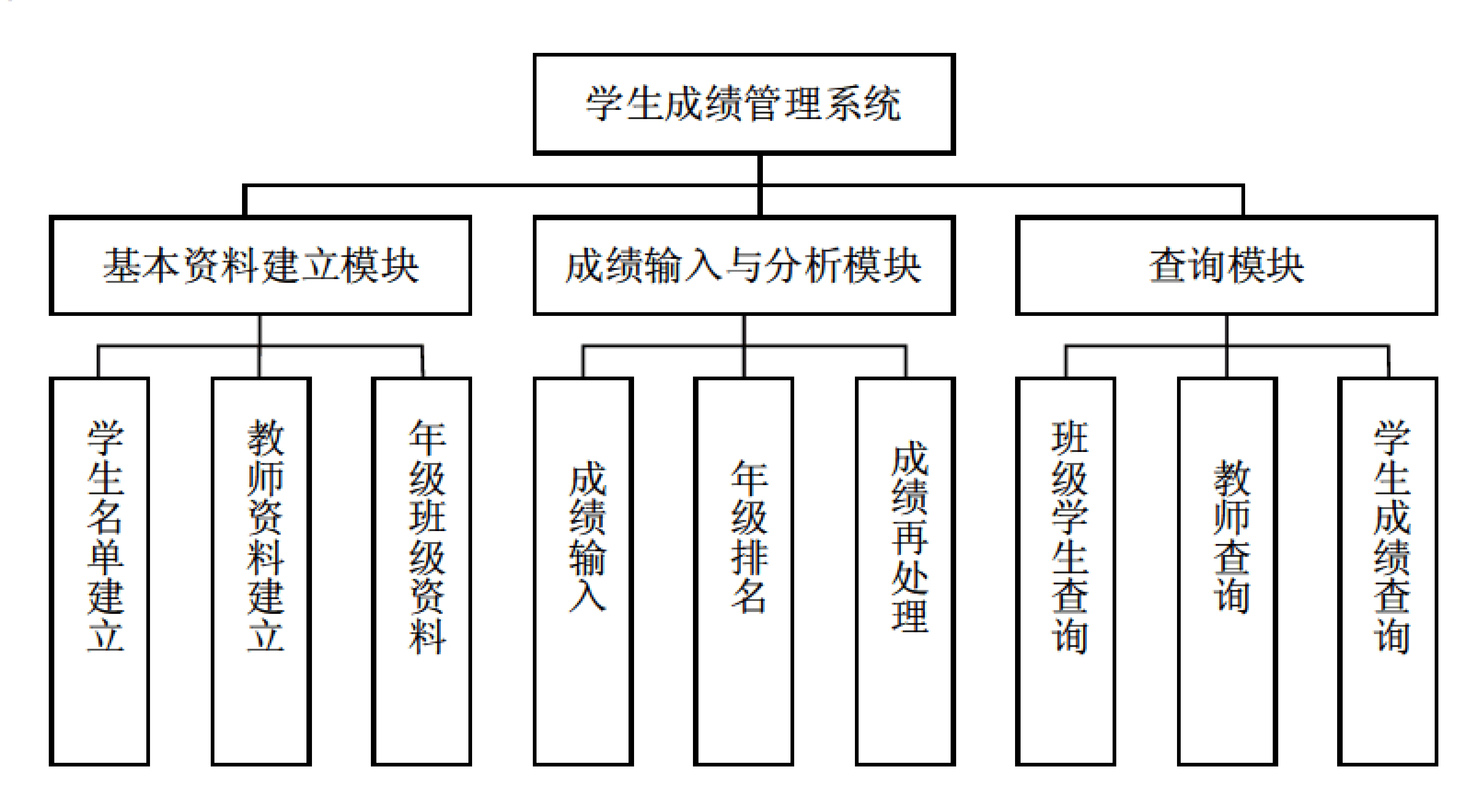


图 5-1　系统详细结构图

以下为各个模块的详细功能介绍。

- 基本资料建立模块：该模块完成系统基本资料的建立。包括学生名单建立、教师资

料建立与年级班级资料建立 3 个子功能块。

- 成绩输入与分析模块：该模块完成学生成绩资料的输入与分析工作，包括成绩输入、年级排名与成绩再处理 3 个子功能块。
- 查询模块：该模块完成学生、老师资料与成绩查询工作，包括班级学生查询、教师查询与学生成绩查询 3 个子功能块。

本例大量使用了 Excel 自带的功能，例如排序、自动筛选、数据有效性。排序功能主要使用在对学生成绩表的名次计算中，首先对总分按照降序排列，然后按照顺序计算名次。自动筛选功能，主要使用在各个查询模块，结合窗体获取筛选条件，然后使用自动筛选产生查询结果。数据有效性功能，主要使用在需要输入年级与班级的输入工作表中，以确保输入的年级与班级都是在年级与班级设置表中存在的，从而保证数据输入的正确性。

在成绩输入与分析模块中的成绩输入功能模块中，设计了求总分、计算班级名次以及保存班级成绩数据表功能。对于已保存的班级成绩数据表，总分和名次是静态数据。当需要重新修改时需要重新将数据返回成绩输入界面，所以设计了成绩再处理功能模块。使使用者能针对需要对成绩做出修改，然后重新产生总分与班级名次。

5.1.2 知识点一：数据有效性

数据有效性设置可以通过菜单选择也可以通过 VBA 来控制。使用菜单操作时，首先选择【数据】菜单，在【数据工具】区域选择【数据有效性】命令（如图 5-2 所示），然后在弹出的下拉菜单中选择【数据有效性】命令（如图 5-3 所示），此时即可在弹出的【数据有效】性窗口中进行有效性设置（如图 5-4 所示）。

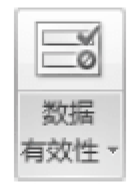


图 5-2 【数据有效性】菜单

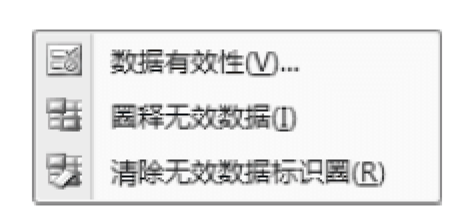


图 5-3 【数据有效性】菜单项

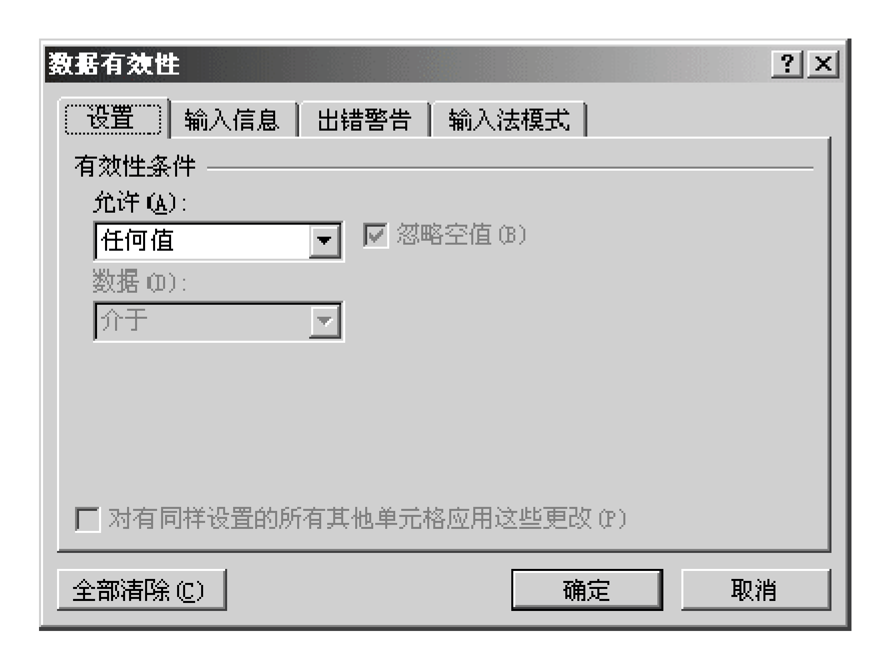


图 5-4 设置数据有效性

数据有效性的 VBA 对象是 Validation，该对象常用的方法是 Add，下面是其具体应用的一个 VBA 代码实例：

```
Range("e5").Validation .Add Type:=xlValidateList, _          '设置数据有效性类型（必需）
        AlertStyle:=xlValidAlertStop, _                     '设置有效性检验警告样式（可选）
```

```
Operator:= xlBetween, _                    '设置数据有效性运算符（可选）
Formula1:="1,2,3,4,5,6,7,8"                '设置数据有效性公式代码说明：
```

Add 方法中只有 Type 属性是必需的，它指定数据有效性的类型，对应于设置窗口的“允许”下拉列表框，例如上面实例中设置的序列形式 xlValidateList。该属性可取值还有 xlValidateCustom（自定义）、xlValidateDate（日期）、xlValidateDecimal（小数）、xlValidateInputOnly（任意值）、xlValidateTextLength（文本长度）、xlValidateTime（时间）、xlValidateWholeNumber（整数）。

AlertStyle 参数用来设置有效性检验警告样式，其取值可以为 xlValidAlertInformation、xlValidAlertStop 或 xlValidAlertWarning。

Operator 参数用来设置数据有效性运算方式，对应于设置窗口的“数据”下拉列表框。对于序列形式，它是固定的，即 xlBetween（介于），其可取值可以为 xlBetween（介于）、xlEqual（等于）、xlGreater（大于）、xlGreaterEqual（大于等于）、xlLess（小于）、xlLessEqual（小于等于）、xlNotBetween（未介于）或 xlNotEqual（不等于）。

Formula1 参数用来设置数据有效性公式的第一部分。如果设置的是字符串序列，则使用“,”隔开序列值。

5.1.3　知识点二：自动筛选

自动筛选是 Excel 2007 自带的强大的数据查询与分析功能。通过该功能，对每一个需要筛选的数据列，都可以设置 3 个以内的筛选条件。使用该功能也可以采用菜单操作和 VBA 操作两种方式。

使用菜单操作时，首先选中需要自动筛选的列（如果不选择，Excel 2007 会自动确定需要筛选的列）。然后选择【数据】菜单，在【排序和筛选】中单击【筛选】按钮（如图 5-5 所示）。此时该数据列将开启自动筛选，并且标题单元格右下角多了一个下拉箭头。单击该下拉箭头即可设置自动筛选（如图 5-6 所示）。



图 5-5　【筛选】按钮

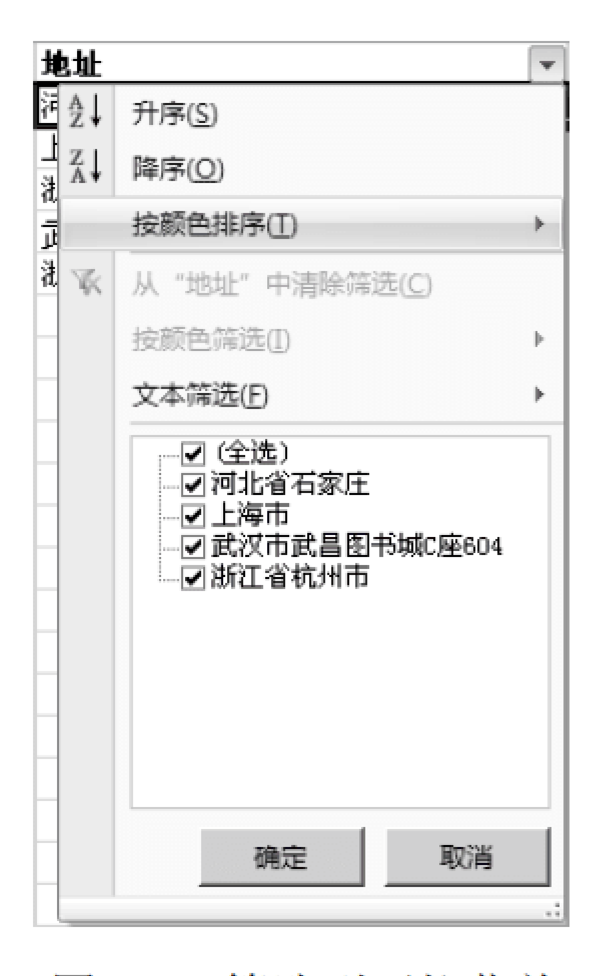


图 5-6　筛选列下拉菜单

图中前面 3 个选项是用来设置排序方式的。单击【文本筛选】按钮后弹出其二级菜单（如图 5-7 所示），在此选择对应的筛选条件方式。选择【自定义筛选】命令后弹出【自定义自动筛选方式】对话框（如图 5-8 所示）。也可以在如图 5-6 所示的复选框中选中需要筛选的值。



图 5-7　文本筛选二级菜单

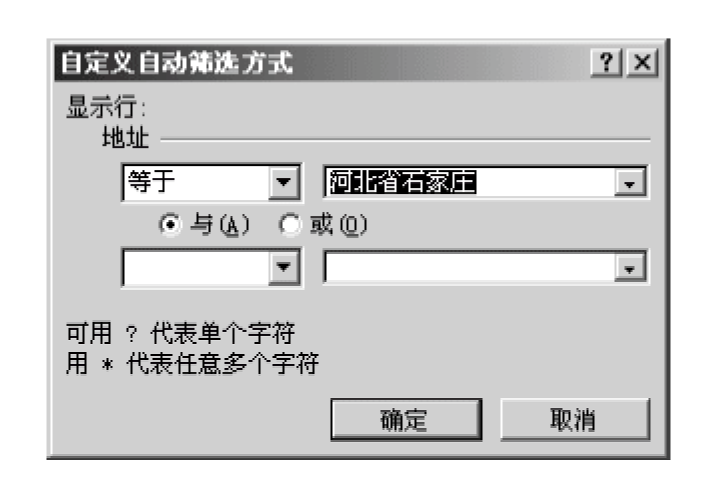


图 5-8　自定义自动筛选方式

下面是一个 VBA 代码实例：

```
With Columns("A:D")
    .AutoFilter                                    '开启自动筛选
    .AutoFilter Field:=1, Criteria1:="SS"          '筛选 A 列中等于"SS"的单元格
    .AutoFilter Field:=4, Criteria1:="=*Auto *", _ '筛选 D 列中包含"Auto"字符串的单元格
        Operator:=xlAnd
End With
```

代码首先对当前表的 A 到 D 列开启自动筛选，然后设置第一列（A 列）筛选条件为：等于“SS”的项目；设置第四列（D 列）筛选条件为：包含 Auto 字符串的项目。整个筛选的条件就是：筛选出 A 列中等于“SS”、D 列中包含 Auto 字符串的项目。

Operator 标记筛选的运算方式。它的取值为 xlAnd（和，该运算方式为默认运算方式）、xlBottom10Items（后 10 项）、xlBottom10Percent（后 10%项）、xlOr（或）、xlTop10Items（前 10 项）、xlTop10Percent（前 10%项）。

5.1.4　知识点三：冻结窗口

冻结窗口可以固定表中的某些列或某些行的数据，保证它不会同步于滚动条的滚动。这样可以方便浏览数据。比如通常把表的标题列固定下来，以便于向下浏览数据时，标题列一直存在。

手动操作冻结窗口时，首先选中某个单元格，然后选择【视图】菜单，在【窗口】区域中单击【冻结窗口】按钮（如图 5-9 所示），弹出【冻结窗口】的二级菜单（如图 5-10 所示），这里选择【冻结拆分窗格】选项。此时，选择的单元格上部与左部的区域为固定区域，该单元格下部以及右部（包括本身）为可滚动区域。

【冻结窗口】的二级菜单中其他两个按钮分别用来将首行和首列冻结，是一种快捷的冻结标题行和标题列的方式。

选择 A2 单元格，然后选择【冻结拆分窗口】选项和直接选择【冻结首行】选项的结果一致。选择 B1 单元格，然后选择【冻结拆分窗口】选项和直接选择【冻结首列】选项的结果一致。

冻结窗口后，再次进入【冻结窗口】菜单时，【冻结拆分窗口】已变成【取消冻结窗格】，

再选择该选项可以取消冻结窗口。



图 5-9　【冻结窗口】按钮

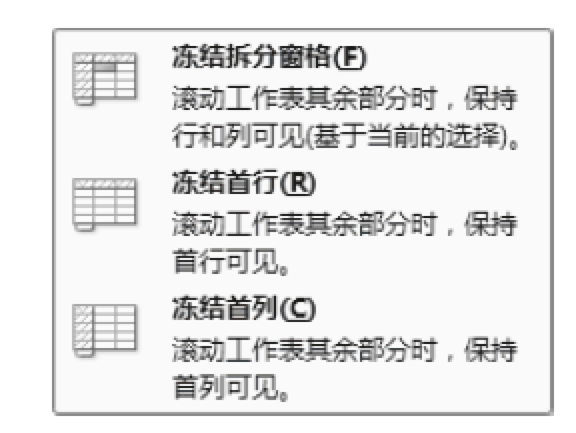


图 5-10　冻结窗口二级菜单

冻结窗口的 VBA 代码如下：

```
ActiveWindow.FreezePanes = True
```

5.1.5　知识点四：End 属性

使用单元格的 End 属性将会得到一个单元格区域对象。这个对象代表包含源单元格区域的区域尾端的单元格。它等同于在 Excel 中使用 Ctrl+方向键（包括向上键、向下键、向左键和向右键）的功能。下面是该属性 VBA 使用的一个实例：

```
Worksheets("Sheet1").Activate                    '激活 Sheet1
Range("B4").End(xlToRight).Select                '从 B4 单元格向右跳到包含 B4 的非空区域的尾部
```

该示例首先激活 Sheet1 表，然后自 B4 单元格向右定位到包含 B4 的非空区域的尾端。End 的方向参数常量有 xlDown（向下键），xlToRight（向右键），xlToLeft（向左键），xlUp（向上键）。以下代码通常用来获取表格中有数据区域的最大行数。

```
.Cells(Rows.Count,1).End(xlup).Row
```

5.1.6　知识点五：Sort 方法

Sort 方法可以对某列或多个列按指定方式进行排序。在 Excel 2007 的【数据】菜单栏中可以找到【排序】按钮，该按钮可用来手动完成排序设置。排序也可以通过代码完成，该代码的参数众多，但是所有参数都是可选的。以下是该方法的语法格式：

```
Range.Sort(Key1, Order1, Key2, Type, Order2, Key3, Order3, Header, OrderCustom, MatchCase,
Orientation, SortMethod, DataOption1, DataOption2, DataOption3)
```

- ❑ Key1 参数：指定第一排序字段，作为区域名称（字符串）或 Range 对象；确定要排序的值。
- ❑ Order1 参数：确定 Key1 中指定的值的排序次序。
- ❑ Key2 参数：第二排序字段；对数据透视表进行排序时不能使用。
- ❑ Type 参数：指定要排序的元素。
- ❑ Order2 参数：确定 Key2 中指定的值的排序次序。
- ❑ Key3 参数：第三排序字段；对数据透视表进行排序时不能使用。

- Order3 参数：确定 Key3 中指定的值的排序次序。
- Header 参数：指定第一行是否包含标题信息。xlNo 是默认值；如果希望 Excel 确定标题，则指定 xlGuess。
- OrderCustom 参数：指定在自定义排序次序列表中的基于 1 的整数偏移。
- MatchCase 参数：设置为 True，执行区分大小写的排序；设置为 False，则执行不区分大小写的排序。不能用于数据透视表。
- Orientation 参数：指定以升序还是降序排序。
- SortMethod 参数：指定排序方法。
- DataOption1 参数：指定 Key1 中所指定区域中的文本的排序方式。不能用于数据透视表排序。
- DataOption2 参数：指定 Key2 中所指定区域中的文本的排序方式。不能用于数据透视表排序。
- DataOption3 参数：指定 Key3 中所指定区域中的文本的排序方式。不能用于数据透视表排序。

5.2 首 页 设 计

本例的首页使用形状图形进行跳转工作，单击相应功能块的自选图形就会执行相应的跳转子模块。界面中共包含了 10 个形状图形，其中有 9 个圆角矩形作为跳转按钮，还有 1 个矩形则作为整个首页的外边框。外边框中还包含了 3 个分组框。这 3 个分组框分别用于分隔开不同功能分类的按钮。该首页的界面效果如图 5-11 所示。



图 5-11 首页界面效果图

首页中不包含任何代码，所以本部分将该界面的建立步骤不再单独列一小节加以介绍。以下是该界面的具体建立步骤：

（1）建立界面外边框。在 Excel 2007 中依次选择【插入】|【形状】|【矩形】|【矩形】命令。在首页的空白区域单击鼠标左键并按住不放向右下方拖动，到适当大小时释放鼠标，即可生成一个矩形外边框。该矩形将作为整个首页界面的外框。

（2）设置界面外边框。右击矩形外边框，在弹出的快捷菜单中选择【编辑文字】命令，如图 5-12 所示。然后输入文字内容为“学生成绩管理系统”。

（3）选中在矩形外边框中输入的文字并右击。在随后弹出的快捷菜单中选择【字体】命令，如图 5-13 所示。此时将会打开【字体】对话框，如图 5-14 所示。在【字体】选项卡中将字体设置为“华文彩云”，字号为 20。然后再次右击矩形外边框，在弹出的快捷菜单中选择【设置形状格式】菜单项，如图 5-12 所示。在【设置形状格式】窗口中选择【文本框】项目，然后设置垂直对齐方式为“顶端对齐”，如图 5-15 所示。



图 5-12　编辑形状文字



图 5-13　设置字体

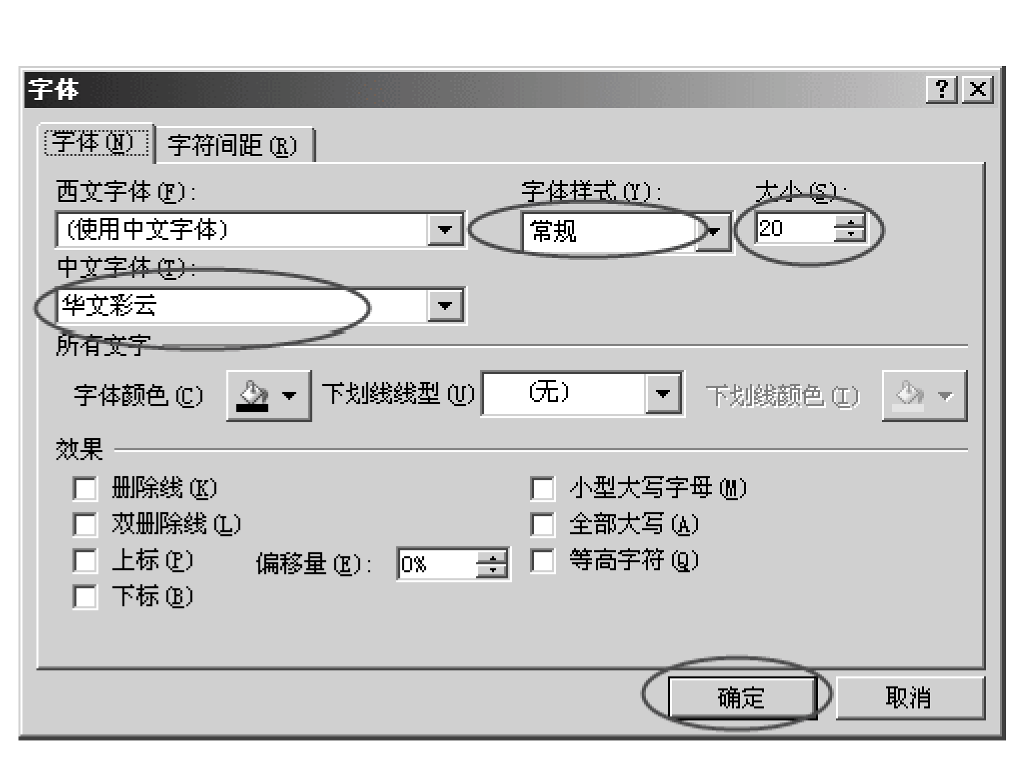


图 5-14　设置界面外边框字体属性

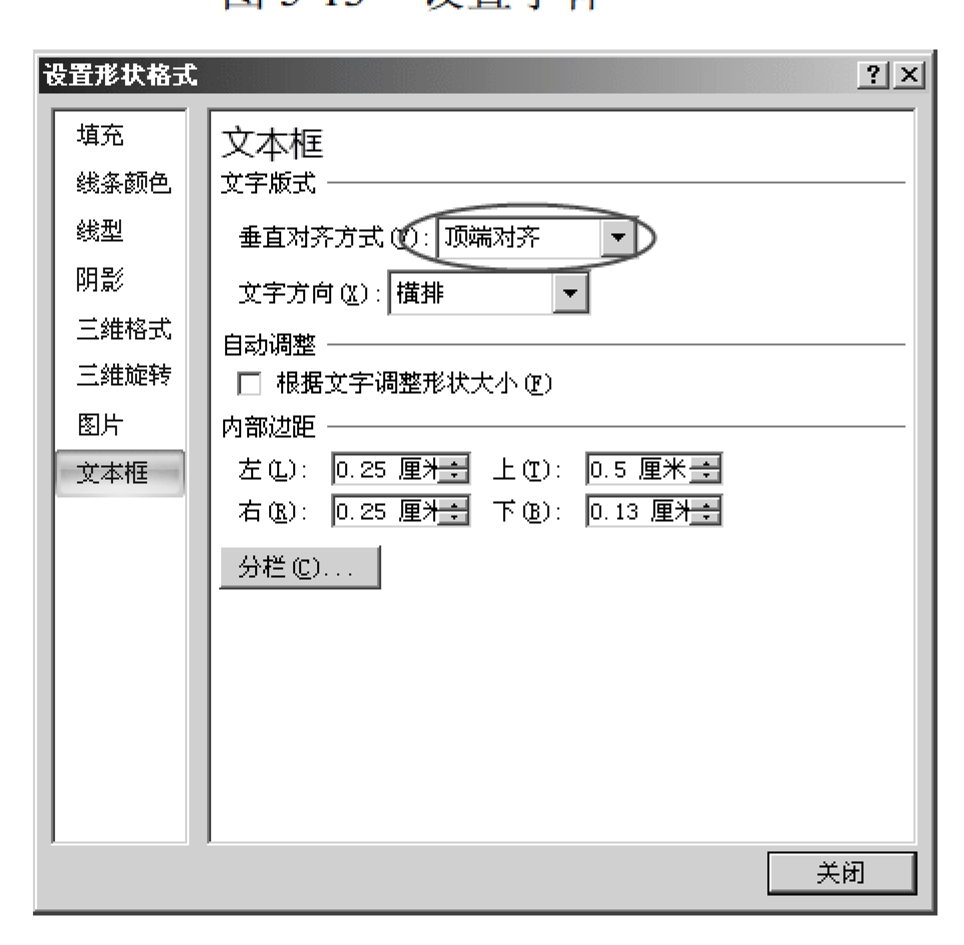


图 5-15　设置界面外边框对齐属性

（4）建立分组框。在 Excel 2007 中依次选择【开发工具】|【插入】|【表单控件】|【分组框】命令。在第 4 章中已经介绍了如何调出【开发工具】选项卡，这里不再说明。随后在系统首页界面矩形的内部连续建立 3 个分组框，然后分别右击 3 个分组框，在弹出的快捷菜

单中选择【编辑文字】命令，如图 5-12 所示。将 3 个分组框的文字内容分别设置为“基本资料建立”、“成绩输入与分析”和“查询”，如图 5-11 所示。

（5）建立各模块跳转按钮。按钮都是使用形状图形来完成的，下面只讲述第一个按钮“学生名单建立”的制作步骤，其他按钮以此类推。首先在 Excel 2007 中依次选择【插入】|【形状】|【矩形】|【圆角矩形】命令。随后在【基本资料建立】分组框中拖动出一个适当大小的圆角矩形。

（6）设置跳转按钮。右击创建的圆角矩形，在弹出的快捷菜单中选择【编辑文字】命令，如图 5-12 所示。然后输入文字内容为“学生名单建立”。随后再次右击该圆角矩形，在弹出的快捷菜单中选择【设置形状格式】命令，如图 5-12 所示。随后将打开【设置形状格式】对话框。然后在【设置形状格式】对话框中选择【填充】项目，如图 5-16 所示，在填充设置中选中【渐变填充】单选按钮。然后再展开【颜色】下拉列表框并选择【橙黄，强调文字颜色 6，淡色 40%】项目，如图 5-17 所示。

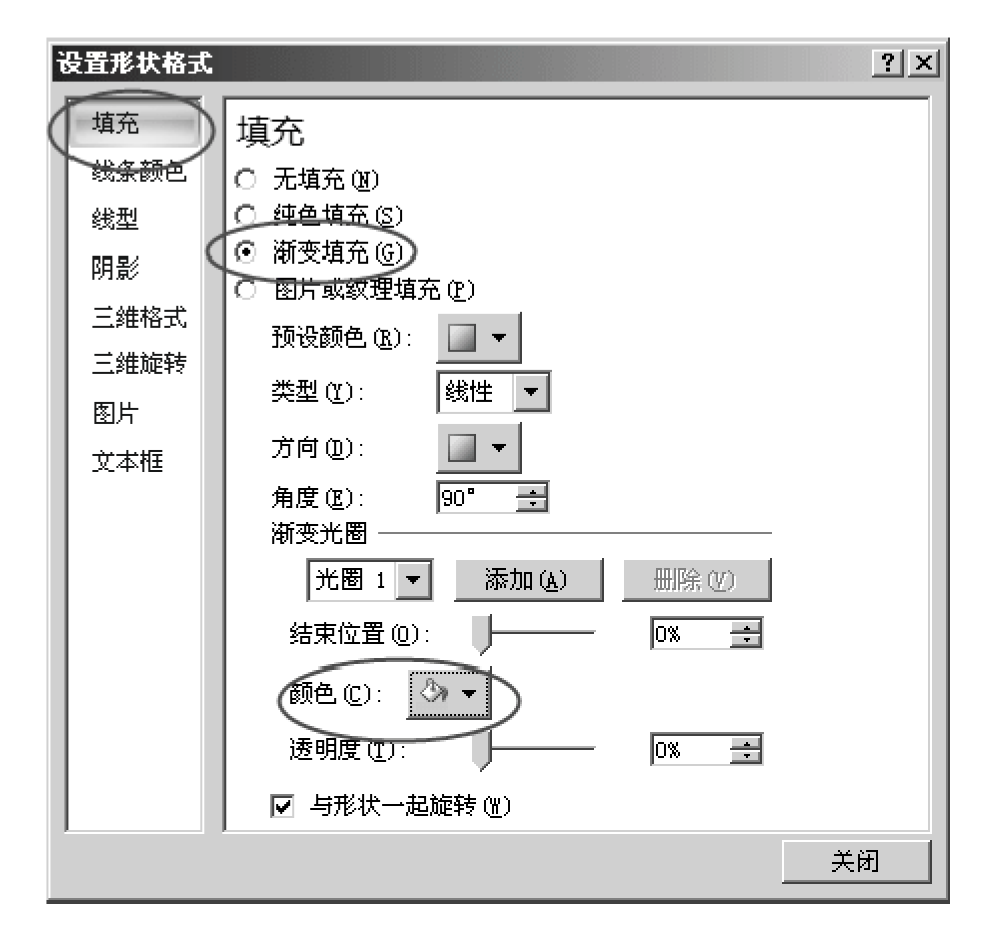


图 5-16　设置填充效果

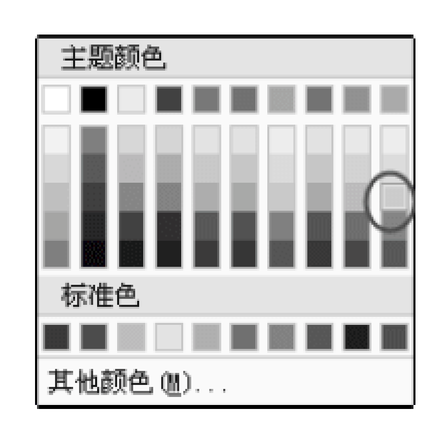


图 5-17　填充颜色设置

（7）设置按钮的宏过程。按钮和相应的过程代码完成后，需要将这些按钮及其单击时相应的过程对应起来。首先右击【学生名单建立】按钮，在弹出的快捷菜单中选择【指定宏】命令，如图 5-18 所示。随后将打开【指定宏】对话框，如图 5-19 所示。在该对话框的【宏名】列表框中选择已经建立好的过程，这里选择已创建好的宏 XSMD 即可。



图 5-18　指定宏快捷菜单



图 5-19　【指定宏】对话框

5.3　基本资料建立模块设计

基本资料建立模块用于完成系统的基础资料信息建立。该模块完成的工作包括学生名单建立，教师资料建立，班级年级资料的建立。这些工作分别由首页基本资料建立分组框中的三个按钮完成。这 3 个子功能模块的描述如下：

- 学生名单建立子模块：该子模块用于建立学生基本信息。单击首页中的【学生名单建立】按钮后，将激活学生名单工作表。该表包含了 4 个限定学生信息的数据列，分别是学号、年级、班级名和学生名。用户建立学生信息时，可以直接在数据末端输入新的学生信息。
- 教师资料建立子模块：该子模块用于建立教师基本信息。单击首页中的【教师资料建立】按钮后，将激活教师资料工作表。该表包含了 11 个数据列，分别是教师名、班主任、语文、数学、英语、政治、生物、物理、化学、历史和地理。用户需要输入的是教师的名称以及相应教授课程下的班级名称。
- 班级年级资料建立子模块；该子模块用于建立学校年级与班级名资料。表中年级名列是非重复的年级名称，而班级名列中存储了所有班级的名称，这些名称还包含了年级信息。

5.3.1　学生名单表设计

学生名单表主要用于管理学生学号、年级、班级名称以及学生名信息。该表用于存储学校所有学生的相关信息。其允许保存的学生数量是 6 万多，对于普通的学校已经够用。学生名单表界面如图 5-20 所示。这些信息被建立后，用户可以通过查询模块查询具体班级或某学号的学生信息。查询学生资料时是采用自动筛选的方式筛选对应的学生学号、年级、班级名称、学生名列。查询的功能请参见后续小节的介绍。

学生成绩管理系统-1 (version 1).xlsb [自动保存的]

	A	B	C	D	E	F	G	H	I
1	学号	年级	班级名	学生名				返回	
2	21054	初二年级	初二年级1班	明春					
3	21053	初二年级	初二年级1班	任视正					
4	21052	初二年级	初二年级1班	覃崇望					
5	21051	初二年级	初二年级1班	覃贵婉					
6	21050	初二年级	初二年级1班	覃　偉					
7	21049	初二年级	初二年级1班	韦孟云					
8	21048	初二年级	初二年级1班	覃　杰					
9	21047	初二年级	初二年级1班	覃立定					
10	21046	初二年级	初二年级1班	覃伙迅					
11	21045	初二年级	初二年级1班	覃　绒					
12	21044	初二年级	初二年级1班	杨　琴					
13	21043	初二年级	初二年级1班	何　朗					
14	21042	初二年级	初二年级1班	吴熙锦					
15	21041	初二年级	初二年级1班	何　宁					

首页 / 教师资料 / 学生名单 / CJ-初二年级1班 / 成绩输

图 5-20　学生名单表界面

当要建立新的学生名单时，用户首先在首页单击【学生名单建立】按钮。单击该按钮时，程序将执行 XSMD 过程。该过程用于激活学生名单表以及对学生表单的年级列的数据有效性

进行初始化设置。下面是设置数据有效性的程序流程。

首先程序从年级班级表中获取所有年级的序列，注意该年级序列是使用逗号相互连接的；然后程序将该需要设置数据有效性的区域的有效性设置清除；最后使用新的有效性序列设置该区域的数据有效性。如图 5-21 所示的是该过程的流程图。

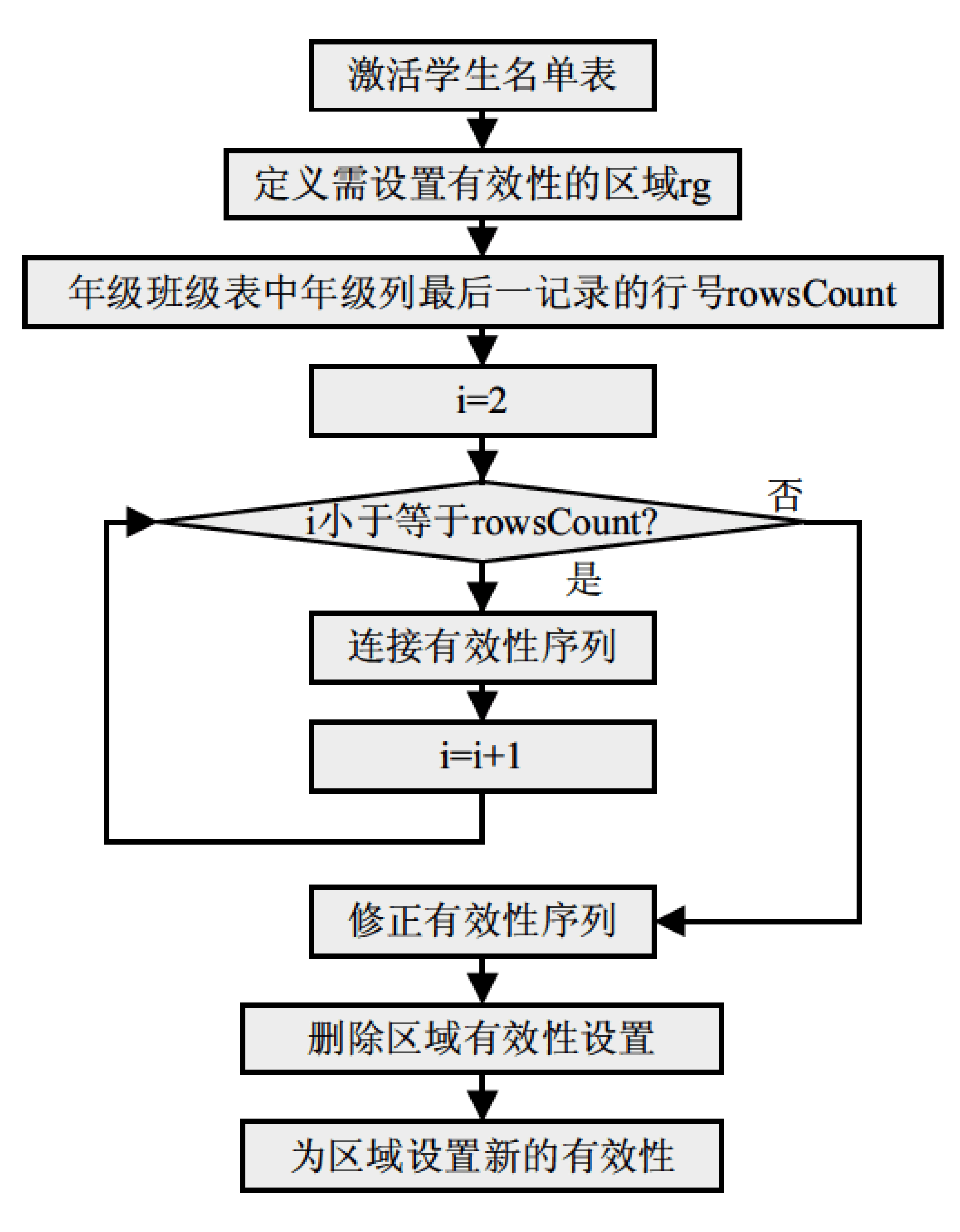


图 5-21　学生名单过程流程图

```
Sub XSMD()
Dim rg As Range, strCell As String, rowsCount As Integer, i As Integer
Sheet2.Activate                                             '激活学生名单表
Set rg = Sheet2.Range(Cells(2, 2), Cells(Rows.Count, 2))    '获取需要设置数据有效性的区域
rowsCount = Sheet6.Cells(Rows.Count, 1).End(xlUp).Row       '获取年级班级表中年级列最后一记录
                                                             的行号
For i = 2 To rowsCount                                      '循环所有年级名称
    strCell = strCell & "," & Sheet6.Cells(i, 1)            '连接年级序列
Next
strCell = Right(strCell, Len(strCell) -1)                   '清除年级序列中多余的逗号
With rg.Validation
    .Delete                                                 '清除区域的数据有效性
    '为区域添加数据有效性
    .Add Type:=xlValidateList, AlertStyle:=xlValidAlertStop, Operator:=xlBetween, Formula1:=strCell
End With
Set rg = Nothing
End Sub
```

上述过程执行完成后，系统当前被激活的工作表即为学生名单表。该表的界面并不复杂，在学生名单表中设置了一个【返回】按钮用于跳转回首页。该按钮的建立类同于前面介绍首页时的跳转按钮。选择该按钮时程序将执行位于“菜单跳转代码”模块的 ReturnMain 过程。


该过程的代码如下：

```
Sub ReturnMain()
Sheet5.Activate                                '激活首页表
End Sub
```

在该表中为了辅助用户快速输入资料，不仅对年级列设置了有效性，而且也为班级列设置了有效性。设置年级列的有效性是在工作表被激活时完成的，而班级列有效性设置是用户修改相应的年级信息时完成的。因而在该工作表中就包含了一个 Worksheet_Change 事件。

程序根据工作表改变事件中获得的 Target 单元格区域对象，判断用户做出的改变是否在工作表的 B 列。当是 B 列时，程序从年级班级表中获取当前选择年级下的所有班级，然后将这些班级连接起来作为一个新的数据有效性序列。随后将该序列作为有效性序列设置给右侧的班级名单元格。该过程的流程如图 5-22 所示。

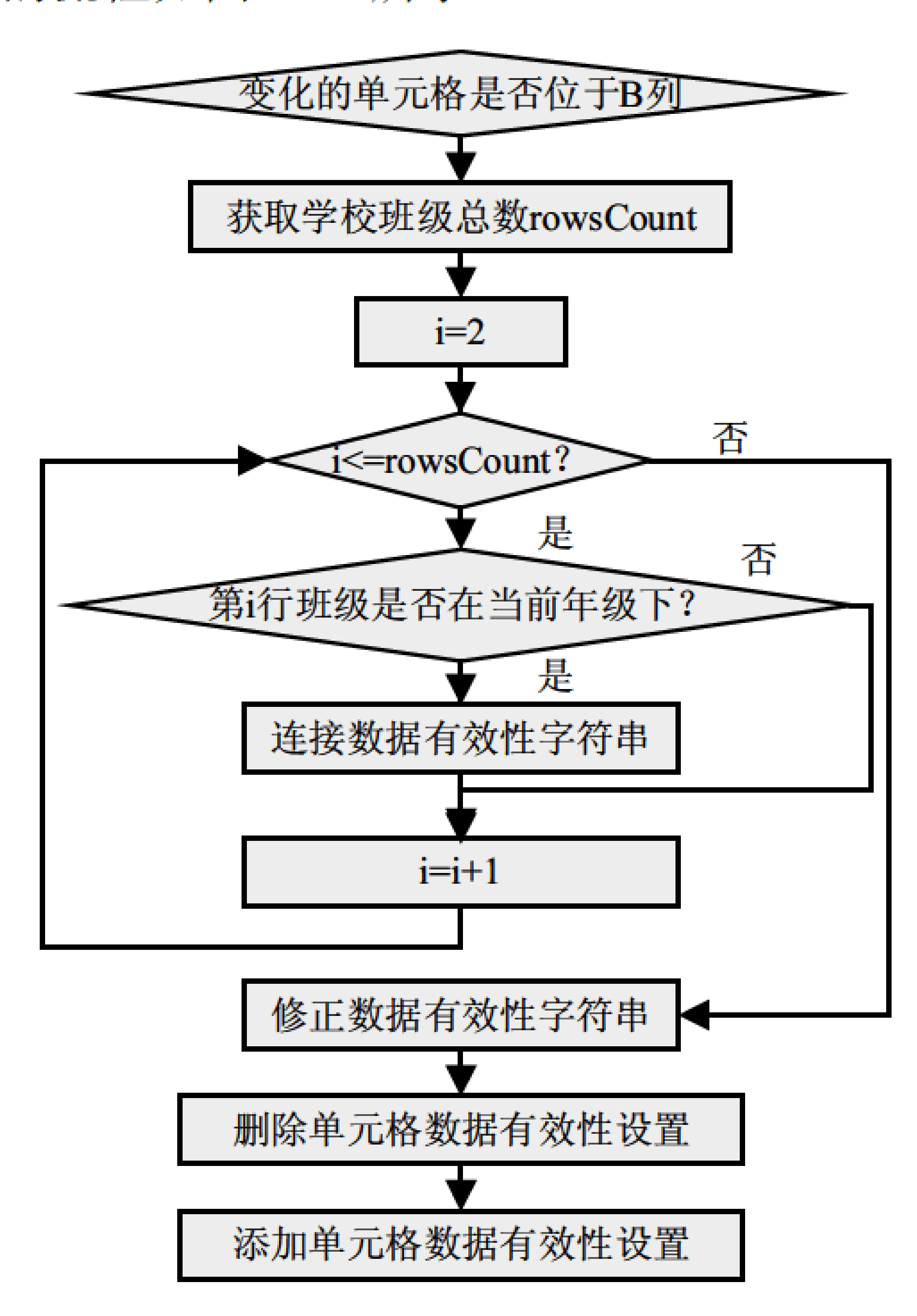


图 5-22　学生名单表改变过程流程图

以下是该过程的代码解释：

```
Private Sub Worksheet_Change(ByVal Target As Range)
Dim rg              As Range              '存储需设置数据有效性的单元格
Dim strCell         As String             '存储筛选条件的字符串
Dim rowsCount       As Integer            '存储最大表行数
Dim i               As Integer            '循环计数变量
On Error GoTo Exit_sub                    '当出现错误时，跳转到过程末尾
If Target.Column = 2 Then                 '当改变的是 B 列数据时，继续执行后续代码
'计算年级班级表最后一个班级数据所在行数
rowsCount = Sheet6.Cells(Rows.Count, 2).End(xlUp).Row
```

```
'将 Target 指定的年级所对应的所有班级序列写入其右边班级单元格数据有效性序列中
For i = 2 To rowsCount
     '检查从年级班级表中获得的班级是否属于当前单元格对应年级
     If Left(Sheet6.Cells(i, 2), Len(Target)) = Target Then
     '将满足条件的班级名写入 strCell 字符串，以备设置数据有效性
          strCell = strCell & "," & Sheet6.Cells(i, 2)
     End If
Next
strCell = Right(strCell, Len(strCell) -1)          '去掉 strCell 最前面多出的分割号
Set rg = Sheet2.Cells(Target.Row, 3)               '定义需要设置数据有效性的单元格
'以下代码参见数据有效性知识点介绍
With rg.Validation
          .Delete                                  '删除单元格有效性设置
          .Add Type:=xlValidateList, AlertStyle:=xlValidAlertStop, _
Operator:=xlBetween, Formula1:=strCell             '允许单元格接受序列中所有值
End With
End If
Set rg = Nothing
Exit_sub:
End Sub
```

5.3.2　教师与科目设置表设计

在首页单击【教师资料建立】按钮后，将会跳转到教师资料表。该表主要用于完成存储教师名称，教师任职科目信息工作。这些建立的信息可以被查询模块调用。该表的界面如图 5-23 所示。



图 5-23　教师资料表界面

表中包含了一个 Worksheet_BeforeDoubleClick 事件，该事件主要用来辅助输入数据。当用户在该工作表中双击输入区的某个单元格时，会弹出一个辅助输入窗口，例如需要设置彭平芳老师担任七年级 2 班的班主任。输入时可以双击 B2 单元格，然后会弹出一个窗口。在窗口的选择年级与选择班级文本框中分别输入年级与班级即可。有关该窗口的详细介绍，请见后续窗口代码设计的年级班级选择窗口设计部分。以下是该工作表的双击事件代码：

```
Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)
'判断是否在表的 B 到 K 列双击，该范围都是可以进行输入班级信息的区域
If Len(Sheet1.Cells(Target.Row, 1)) And Target.Column > 1 And Target.Column < 12 Then
```

```
frmXSMD.Show                                     '显示辅助输入窗口
If Len(Target) Then                              '判断被双击单元格内容是否非空
        Target = Target & " " & tempBJ           '如果非空，将输入结果添加到原内容后
Else
        Target = tempBJ                          '如果为空，直接将结果写入被双击单元格
End If
End If
End Sub
```

5.3.3　年级班级设置表设计

年级班级设置表与学生资料、教师资料表的界面十分类似。该表中包含了两个数据列，分别是年级名和班级名。该首行标题被冻结，当用户在查看或输入年级名与班级名时，首行标题不会移动。另外该表还包含了一个跳转到首页的返回按钮。该表的设置界面如图 5-24 所示。

在该工作表的代码中也应用了工作表的改变事件。该事件用于确认输入的年级名、班级名是否有重复项目，以确保输入数据的正确性。程序根据发生改变单元格所处列号确认是对年级名列还是班级名列进行了重复性检查。对不同列检查的方式是一样的，程序逐个检测该列的各个数据是否与新输入数据一致，当一致时，记录下重复次数。当检查完所有单元格后，如果该重复次数不超过 1，则说明没有重复项目；否则存在重复项目。如图 5-25 所示的是该过程中对年级列进行重复性检查的流程图。

学生成绩管理系统-1

	A	B	C	D	E	F	G
1	年级名	班级名				返回	
2	九年级	九年级1班					
3	初二年级	九年级2班					
4	七年级	九年级3班					
5		九年级4班					
6		九年级5班					
7		初二年级1班					
8		初二年级2班					
9		初二年级3班					
10		初二年级4班					
11		初二年级5班					
12		七年级1班					
13		七年级2班					
14		七年级3班					
15		七年级4班					
16		七年级5班					
17							
18							
19							

首页 / 教师资料 / 学生名单 / CJ-初二年级1班

图 5-24　年级班级设置表界面

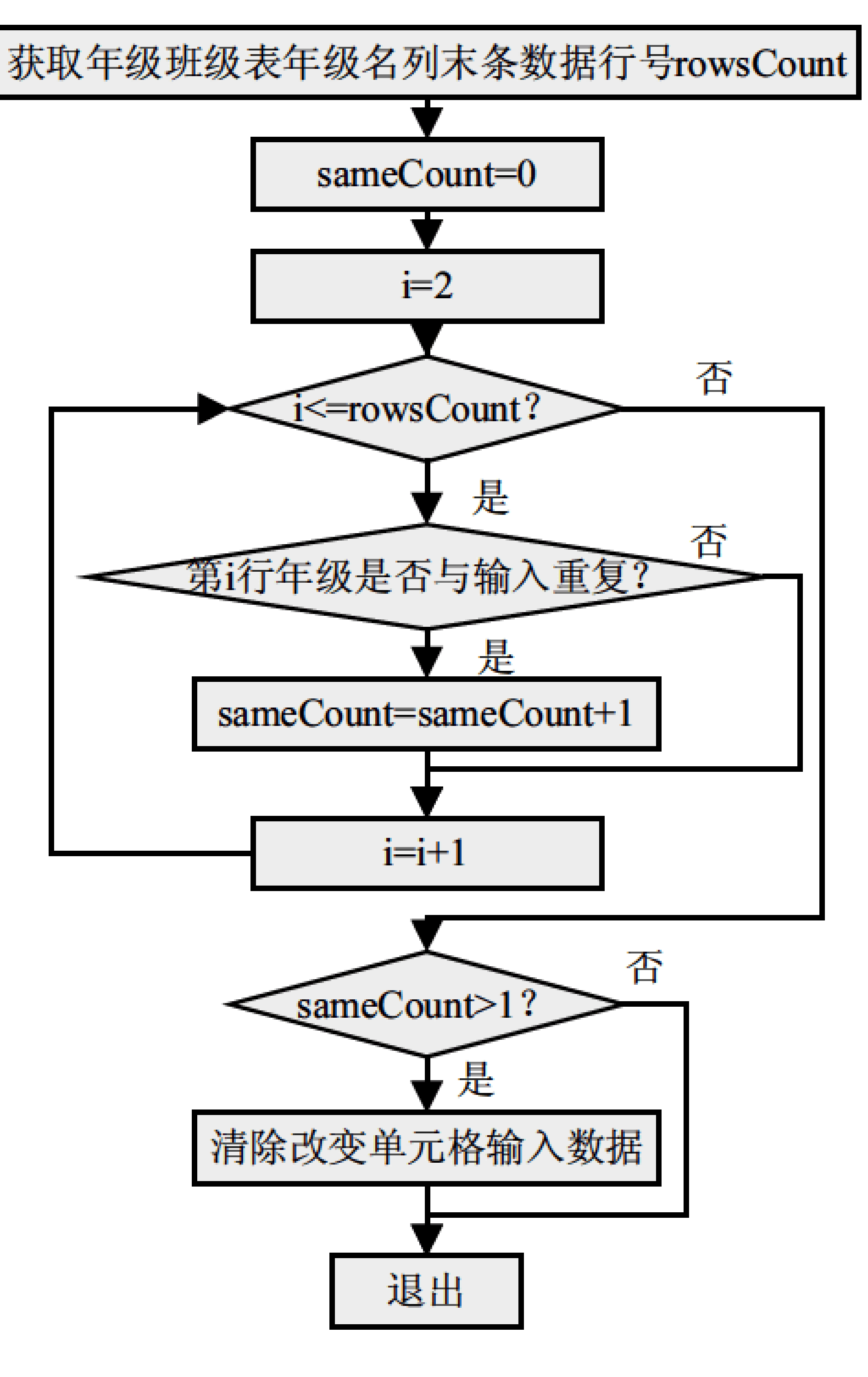


图 5-25　判断年级输入是否重复流程图

以下是工作表改变事件的代码解释：

```
Private Sub Worksheet_Change(ByVal Target As Range)
Dim rowsCount As Integer                    '保存表的最大行数
Dim i As Integer                            '循环计数变量
Dim sameCount As Integer                    '保存班级或年级重复项重复次数
'检查年级输入是否有重复
If Target.Column = 1 Then
    '获取以输入年级数据的最后行的行号
    rowsCount = Sheet6.Cells(Rows.Count, 1).End(xlUp).Row
For i = 2 To rowsCount
    '当检测的单元格与当前变化单元格内容相同时，将计数器增加 1
        If Sheet6.Cells(i, 1) = Target Then
            sameCount = sameCount + 1
        End If
Next
'当计数器大于 1 时（计算时，还计算了本身），可以判断该输入的年级已经存在
    If sameCount > 1 Then
        MsgBox "该年级已经存在！", vbInformation + vbOKOnly
        Target.Clear
    End If
End If
'班级的检测方法同年级的检测方法
If Target.Column = 2 Then
    rowsCount = Sheet6.Cells(Rows.Count, 2).End(xlUp).Row
    For i = 2 To rowsCount
        If Sheet6.Cells(i, 2) = Target Then
            sameCount = sameCount + 1
        End If
    Next
    If sameCount > 1 Then
        MsgBox "该班级已经存在！", vbInformation + vbOKOnly
        Target.Clear
    End If
End If
```

5.4 成绩输入与分析模块设计

成绩输入与分析模块是该实例的重点部分。该模块由 3 部分构成，分别是成绩输入模块、年级排名模块以及成绩再处理模块。在首页表中成绩输入与分析分组框中包含的 3 个按钮分别对应这 3 个功能模块。以下是这 3 个功能模块的功能描述：

- 成绩输入模块：该模块主要完成班级学生成绩的输入工作。用户在首页中单击【成绩输入】按钮后弹出【年级班级选择】对话框。该步用于确认用户输入的年级与班级名。如果用户在此前已经建立了该班级的学生信息，则程序将自动将该班所有学

生信息自动复制到成绩表中。输入完所有的成绩信息后，还可以通过该工作表中的 4 个按钮依次完成相应的功能，分别用于计算学生的总分、计算班级名次、保存成绩表和返回首页。

- 年级排名模块：年级排名模块用于对全年级的学生成绩进行排序。在进行年级学生排名前，需要保证该年级下所有班级的成绩已经建立并且保存到工作簿中。单击该按钮后，用户选择需要统计排名的年级。如果当前选择年级下有班级的成绩信息未建立，程序会提示该班级成绩表未建立并退出统计；否则程序将完成该年级所有学生的排名工作。
- 成绩再处理模块：成绩再处理模块用于再次处理已经保存了的班级成绩表。程序中已经保存了的成绩表没有包含算总分、计算班级名次等功能。这里通过将该班级的数据导入到成绩输入表来完成班级成绩的再处理。当用户需要修改某个学生成绩，然后重新统计总分、排名时，需要通过该操作完成。

5.4.1　成绩输入模块设计

该模块用于建立班级学生成绩。它的主要功能包括班级学生成绩的输入、计算总分、计算班级名次以及保存班级成绩表。该表的界面如图 5-26 所示。该表的界面简洁，几个按钮的建立可以参照首页相关的内容。

学生成绩管理系统-1.xls [兼容模式]

算总分　计算班级名次　保存成绩表　返回

	学号	姓名	语文	数学	英语	政治	生物	物理	化学	历史	地理	总分	班名次
2	21006	蒙　琴	80	70	70	70	70	70	70	100	70	670	1
3	21007	覃院锦	80	70	70	70	70	70	70	99	70	669	2
4	21013	刘振亮	80	70	70	70	70	70	70	70	97.5	668	3
5	21029	覃　键	80	70	70	70	70	70	70	95.5	70	666	4
6	21028	莫交灿	80	70	70	70	70	70	70	95	70	665	5
7	21027	覃亮星	80	70	70	70	70	70	70	94.5	70	665	6
8	21008	覃兴意	80	70	70	70	70	70	70	94	70	664	7
9	21026	蒙碧壮	80	70	70	70	70	70	70	94	70	664	7
10	21025	莫庆山	80	70	70	70	70	70	70	93.5	70	664	8
11	21024	莫旅业	80	70	70	70	70	70	70	93	70	663	9
12	21023	韦覃航	80	70	70	70	70	70	70	92.5	70	663	10
13	21054	柏明春	80	70	70	70	70	70	92.5	70	70	663	10
14	21022	覃万威	80	70	70	70	70	70	70	92	70	662	11
15	21053	任视正	80	70	70	70	70	70	92	70	70	662	11
16	21021	覃春堂	80	70	70	70	70	70	70	91.5	70	662	12
17	21052	覃崇望	80	70	70	70	70	70	91.5	70	70	662	12
18	21020	覃　罗	80	70	70	70	70	70	70	91	70	661	13
19	21051	覃青婉	80	70	70	70	70	70	91	70	70	661	13

首页　教师资料　学生名单　CJ-初二年级1班　成绩输入表　年级排名　年级班级表

图 5-26　成绩输入表界面

单击首页成绩输入与分析分组框中的【成绩输入】按钮后，将调用菜单跳转代码模块的 CJSR 过程。该过程首先打开一个询问年级与班级的对话框（如图 5-27 所示）。该对话框用于设置成绩输入的年级与班级名称。然后将对应的班级的所有学生的学号与姓名写入成绩输入表中，并激活输入工作表。该过程的流程图如图 5-28 所示。

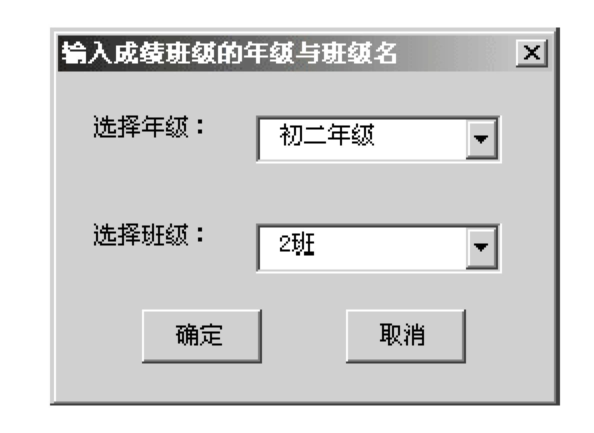


图 5-27　设置输入成绩的年级与班级名

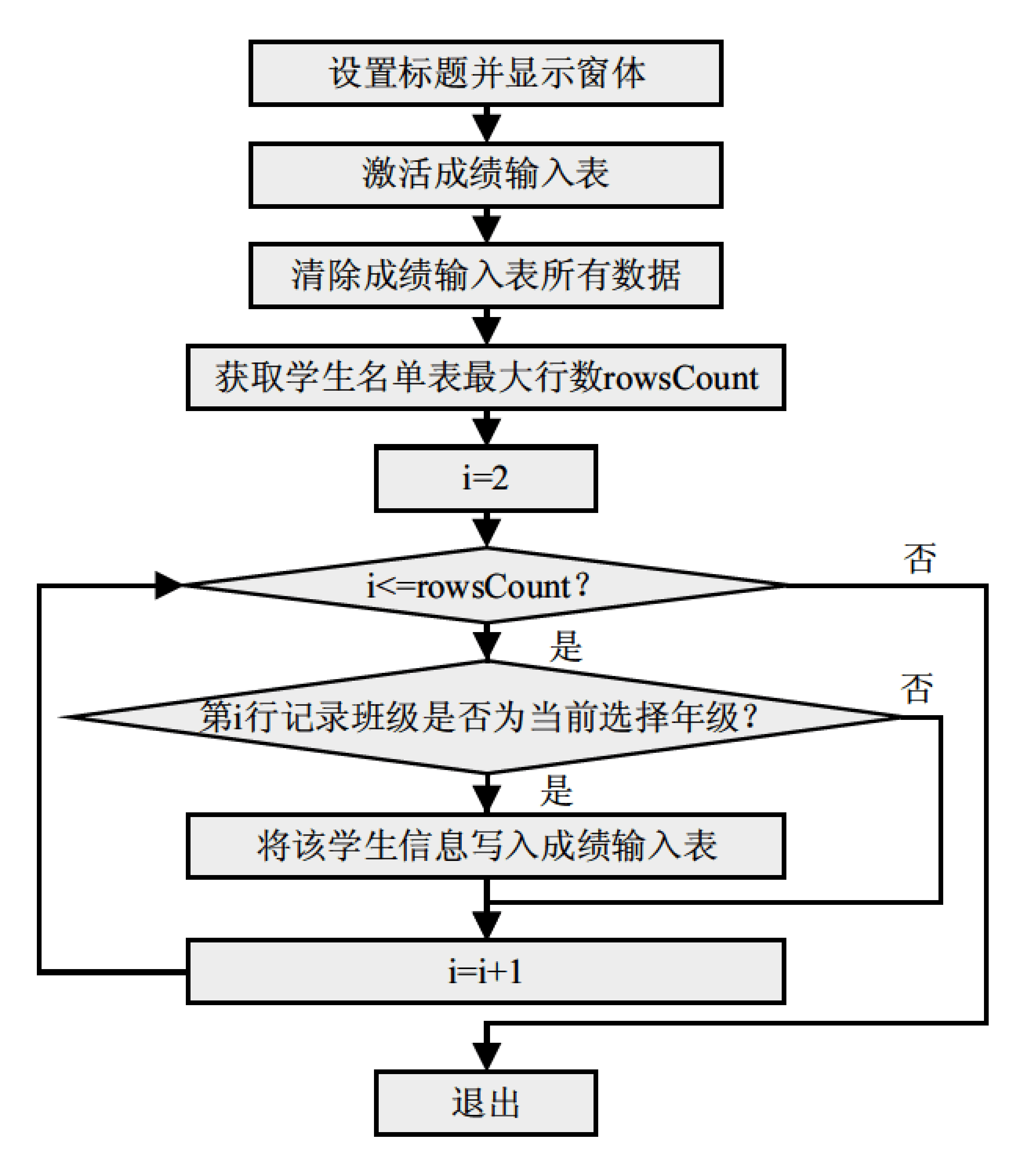


图 5-28　成绩输入过程流程图

该过程的详细代码解释如下：

```
Sub CJSR()
Dim rowsCount As Integer, i As Integer
Dim fillRow As Integer                          '记录已经填充了学号与姓名的最大行数
frmXSMD.Caption = "输入成绩班级的年级与班级名"
frmXSMD.Show                                    '显示获取班级年级名称窗口
Sheet3.Activate                                 '激活成绩输入表
rowsCount = Sheet3.Cells(Rows.Count, 1).End(xlUp).Row               '获取成绩输入表最大行数
If rowsCount > 1 Then                           '判断成绩输入表是否已有成绩数据存在
    Sheet3.Range(Cells(2, 1), Cells(rowsCount, 13)).ClearContents   '清除已有数据
End If
'获取学生名单表最大行数
rowsCount = Sheet2.Cells(Rows.Count, 1).End(xlUp).Row
For i = 2 To rowsCount                          '从学生名单表第二行开始，一直循环到末尾
    If Sheet2.Cells(i, 3) = tempBJ Then         '判断学生所在班级与输入的班级是否对应
        '获取已经写入成绩输入表的学生资料的最大行数
fillRow = Sheet3.Cells(Rows.Count, 1).End(xlUp).Row
        Sheet3.Cells(fillRow + 1, 1) = Sheet2.Cells(i, 1)           '将学生学号写入成绩输入表
        Sheet3.Cells(fillRow + 1, 2) = Sheet2.Cells(i, 4)           '将学生名写入成绩输入表
    End If
Next
End Sub
```

下面依次解释该工作表中包含的几个按钮的功能。该工作表中共包含了 4 个按钮，分别

是算总分、计算班级名次、保存成绩表和返回按钮。返回按钮的功能这里不再加以说明，前面的一些工作表中有类似的按钮。

- 【算总分】按钮：单击该按钮时，程序将循环成绩输入表中的所有数据行将各个学生的总成绩计算出来，然后将该成绩保存到【总分】列中。程序首先获取了成绩输入表末端数据行的行号 rowsCount，然后使用 For 循环从 2 开始一直循环到 rowsCount，将各个科目的成绩相加然后保存到总分列中。该按钮执行宏的流程如图 5-29 所示。
- 【计算班级名次】按钮：单击该按钮后，将执行功能表模块中的 Rank 过程。该过程使用了 Excel 2007 的内置功能 Sort 方法实现对班级学生成绩的排序。关于该方法的知识介绍参见本章的知识点五。程序首先使用 Sort 方法对学生总成绩按照降序排列，然后再将排名列依次从 1 开始编排。该过程的流程图如图 5-30 所示。

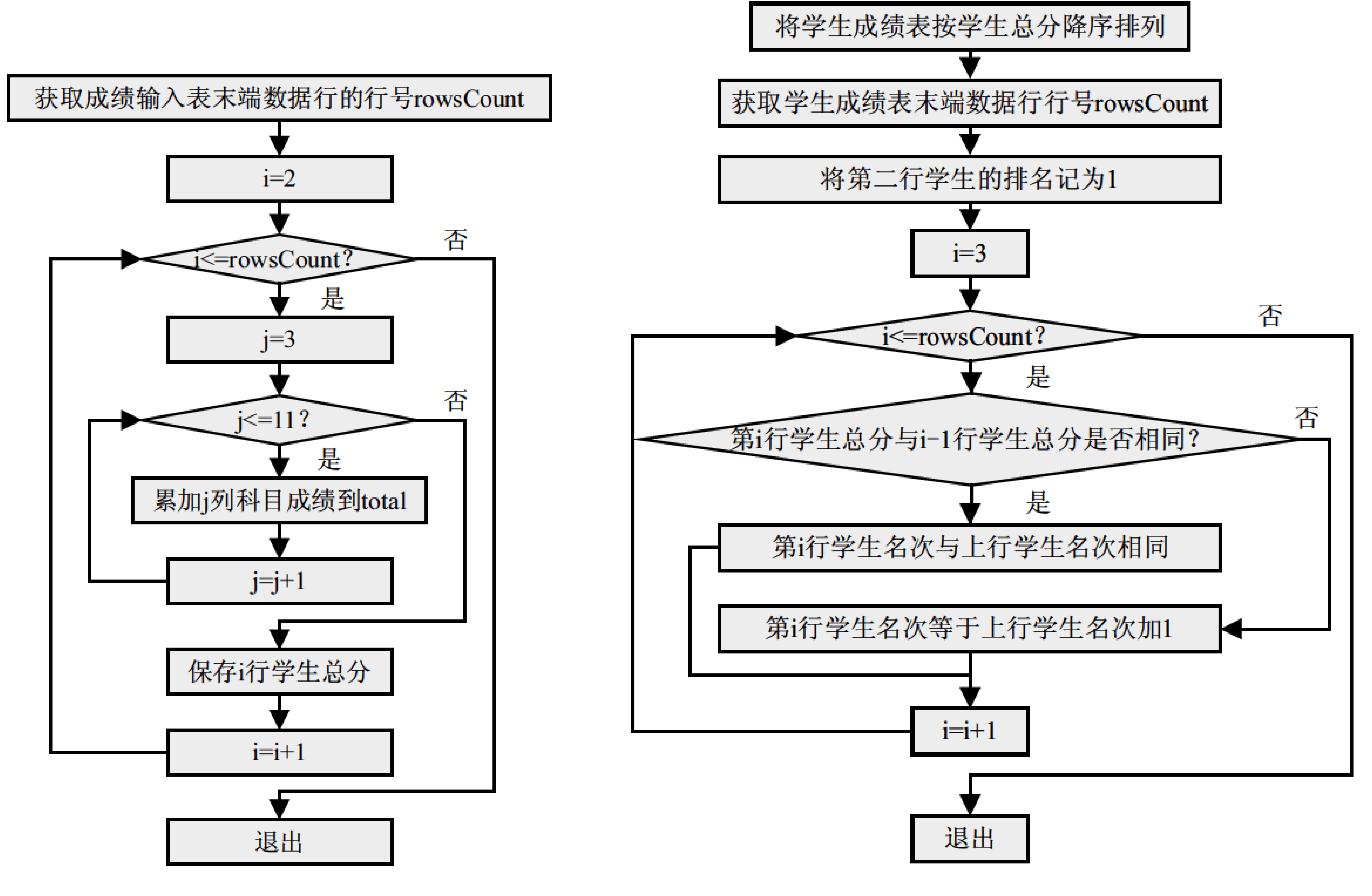


图 5-29　算总分过程流程图　　　　图 5-30　计算学生名次过程流程图

- 【保存成绩表】按钮：单击该按钮后，将会执行功能表模块中的 SaveData 过程。此过程首先会弹出一年级班级输入窗口。用户在该窗口输入成绩表所属年级班级名。当成功获取年级班级名后，过程将新建一个工作表。如果该年级班级的成绩表已经存在，程序首先删除该表，然后再添加新工作表，最后过程将成绩数据复制到新的工作表中。该过程的流程图如图 5-31 所示。

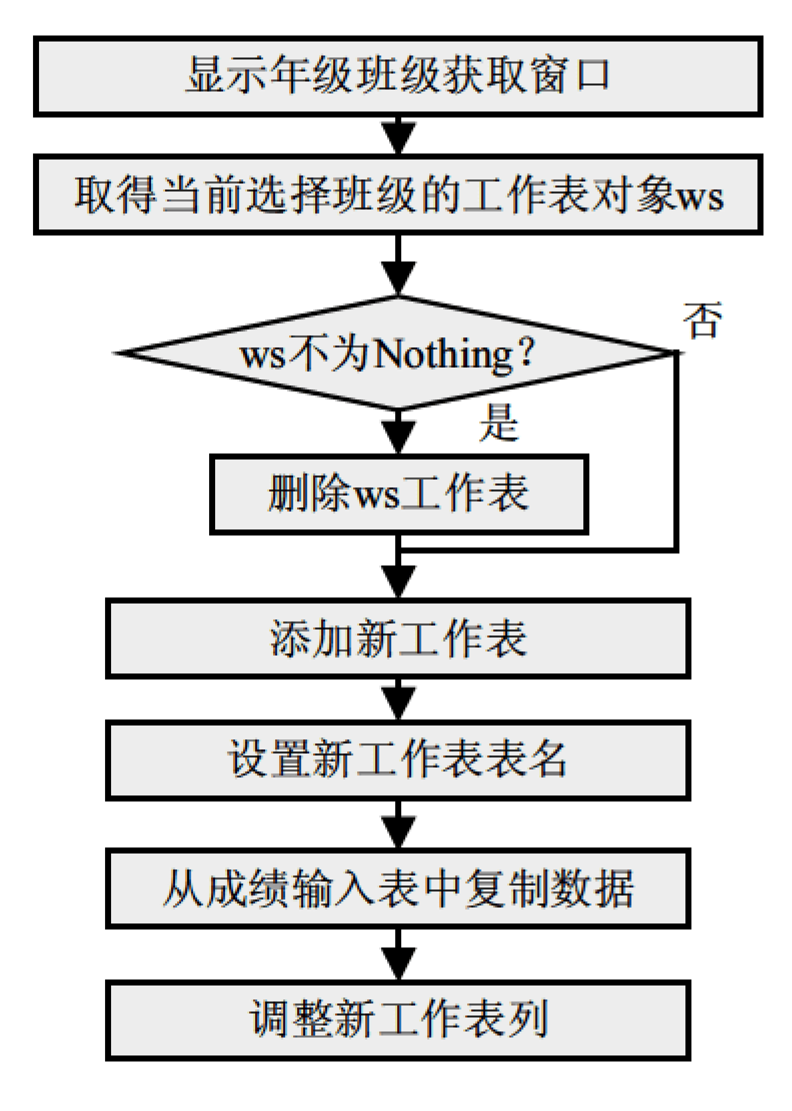


图 5-31　保存成绩过程流程图

以下是算总分按钮的详细代码解释：

```
Sub CalTotal()
Dim rowsCount As Integer                '保存表的最大行数
Dim i As Integer                        '一次循环计数变量
Dim j As Integer                        '二次循环计数变量
Dim total As Double                     '保存总分
'获得成绩输入表当前已建立数据最后一行的行号
rowsCount = Sheet3.Cells(Rows.Count, 1).End(xlUp).Row
'对已输入成绩的行，将各科成绩汇总，然后写入对应的总分列中
For i = 2 To rowsCount
    For j = 3 To 11
        total = total + Sheet3.Cells(i, j)
    Next
    Sheet3.Cells(i, 12) = total
    total = 0                           '在每次计算完一行的总分后，需要将存储总分的临时变量置 0
Next
End Sub
```

以下是计算班级名次按钮的详细代码解释：

```
Sub Rank()
Dim rowsCount As Integer                          '保存表的最大行数
Dim i As Integer                                  '一次循环计数变量
'获得成绩输入表当前已建立数据最后一行的行号
rowsCount = Sheet3.Cells(Rows.Count, 1).End(xlUp).Row
Application.ScreenUpdating = False                '关闭屏幕刷新
'对 A 到 M 列，按照 L 列排序，顺序为降序
Sheet3.Range("A1:M" & rowsCount).Sort Key1:=Range("L2"), Order1:= _
xlDescending, Header:=xlGuess, OrderCustom:=1, _
MatchCase:=False, Orientation:=xlTopToBottom, _
        SortMethod:=xlPinYin, DataOption1:=xlSortNormal
Sheet3.Cells(2, 13) = 1                           '将排到第一位的名次记为 1
```

```
'从第三行开始，将该行的总成绩与上一行的总成绩比对，如果相等，则将其名次设置为上
'一行相同名次,否则将上一行名次加 1 后设置为当前学生的班级名次
For i = 3 To rowsCount
    If Sheet3.Cells(i -1, 12) <> Sheet3.Cells(i, 12) Then
        Sheet3.Cells(i, 13) = Sheet3.Cells(i -1, 13) + 1
    Else
        Sheet3.Cells(i, 13) = Sheet3.Cells(i -1, 13)
    End If
Next
Application.ScreenUpdating = True               '恢复屏幕刷新
End Sub
```

以下是保存成绩表按钮的详细代码解释：

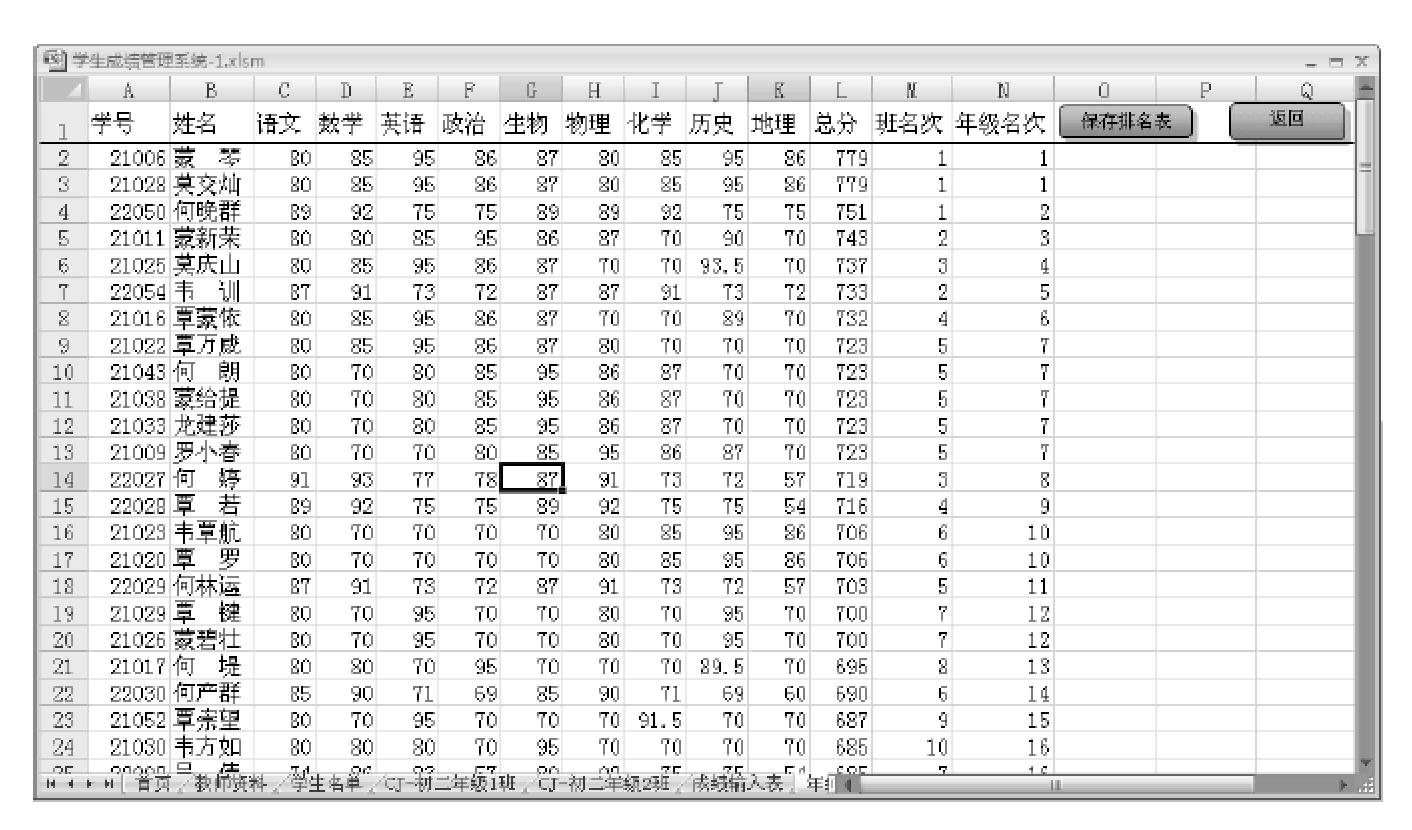
```
Sub SaveData()
Dim ws As Worksheet                             '用于指向保存的成绩表对象
Dim rowsCount As Integer                        '保存表的最大行数
With frmXSMD                                    '显示班级年级获取窗口，并初始化标题
    .Caption = "选择班级名"
    .Show
End With
'取得成绩输入表的最大行数
rowsCount = Sheet3.Cells(Rows.Count, 1).End(xlUp).Row
Application.ScreenUpdating = False              '关闭屏幕刷新
On Error Resume Next
Set ws = Sheets("CJ-" & tempBJ)                 '取得该班级的成绩表对象
If ws Is Nothing Then                           '检测表对象是否获取成功
    Set ws = ThisWorkbook.Sheets.Add            '未获取成功时，说明该班级成绩表未建立
Else                                            '此时新增一个表对象
    ws.Delete                                   '获取成功，说明原来已保存过该表
    Set ws = ThisWorkbook.Sheets.Add            '删除该表，然后新增一个表对象
End If
ws.Name = "CJ-" & tempBJ                        '修改新得到的表对象名称
'从成绩输入表中复制数据到该表对象
Sheet3.Range("A1:M" & rowsCount).Copy ws.Range("A1:M" & rowsCount)
ws.Columns.AutoFit                              '自动调整该表对象的列宽度
Application.ScreenUpdating = True               '开启屏幕刷新
End Sub
```

5.4.2　年级排名模块设计

年级排名模块用于产生某个年级全体学生的总分名次列表，该表只设计了保存功能。如果需要修改该年级的总分名次列表，首先需要修改对应学生所在班级的班级成绩表，然后再重新生成该年级的总分名次列表。

用户在首页单击【年级排名】按钮后，将会弹出【年级选择】对话框。该对话框借用了年级班级选择窗口，用户在这里选择完年级就可以进入排名工作。然后程序会检测当前选择年级下是否所有班级的班级成绩表已经建立。如果存在没有建立的，将提示建立对应班级成绩表；如果所有班级的工作表已经建立，程序自动完成排序工作，并将结果以表列形式显示

出来。该表的界面如图 5-32 所示。该表的显示结果是初二年级的排名，其中初二年级只有两个班级。



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	学号	姓名	语文	数学	英语	政治	生物	物理	化学	历史	地理	总分	班名次	年级名次	保存排名表		返回
2	21006	蒙　琴	80	85	95	86	87	80	85	95	86	779	1	1			
3	21028	莫交灿	80	85	95	86	87	80	85	95	86	779	1	1			
4	22050	何晓群	89	92	75	75	89	89	92	75	75	751	1	2			
5	21011	蒙新荣	80	80	85	95	86	87	70	90	70	743	2	3			
6	21025	莫庆山	80	85	95	86	87	70	70	93.5	70	737	3	4			
7	22054	韦　训	87	91	73	72	87	87	91	73	72	733	2	5			
8	21016	覃蒙依	80	85	95	86	87	70	70	89	70	732	4	6			
9	21022	覃万威	80	85	95	86	87	80	70	70	70	723	5	7			
10	21043	何　朗	80	70	80	85	95	86	87	70	70	723	5	7			
11	21038	蒙给提	80	70	80	85	95	86	87	70	70	723	5	7			
12	21033	龙建莎	80	70	80	85	95	86	87	70	70	723	5	7			
13	21009	罗小春	80	70	70	80	85	95	86	87	70	723	5	7			
14	22027	何　婷	91	93	77	78	87	91	73	72	57	719	3	8			
15	22028	覃　若	89	92	75	75	89	92	75	75	54	716	4	9			
16	21023	韦覃航	80	70	70	70	70	80	85	95	86	706	6	10			
17	21020	覃　罗	80	70	70	70	70	80	85	95	86	706	6	10			
18	22029	何林运	87	91	73	72	87	91	73	72	57	703	5	11			
19	21029	覃　键	80	70	95	70	70	80	70	95	70	700	7	12			
20	21026	蒙碧壮	80	70	95	70	70	80	70	95	70	700	7	12			
21	21017	何　堤	80	80	70	95	70	70	70	89.5	70	695	8	13			
22	22030	何产群	85	90	71	69	85	90	71	69	60	690	6	14			
23	21052	覃荣望	80	70	95	70	70	70	91.5	70	70	687	9	15			
24	21030	韦方如	80	80	80	70	95	70	70	70	70	685	10	16			

图 5-32　年级排名表界面

该工作表相关代码包括两个过程，分别是单击首页中【年级排名】按钮时的宏过程、表中保存排名表按钮宏过程。由于两个过程的代码都比较复杂，这里逐个介绍两过程的功能和代码。

在首页单击【年级排名】按钮后，将调用菜单跳转代码模块的 NJPM 过程。该过程完成询问年级、检测对应年级下所有班级的成绩表是否已经建立以及计算年级排名工作。该过程的执行流程在前面已经有了文字说明，这里给出该过程的流程图。由于该过程比较复杂，将分为 3 个流程图加以说明（如图 5-33~图 5-35 所示），后面两个流程图是第一个流程图步骤之一。

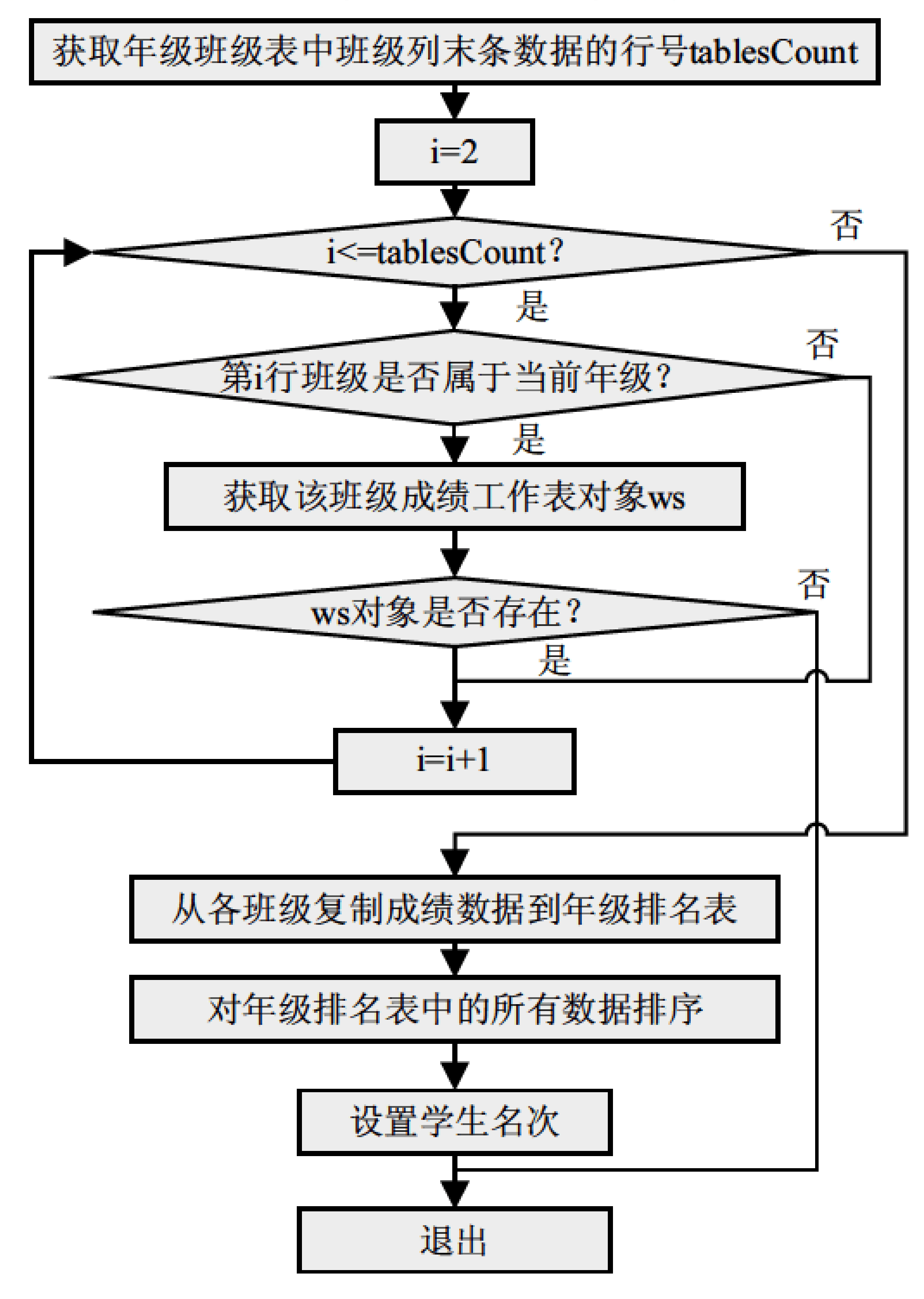


图 5-33　年级排名过程流程图

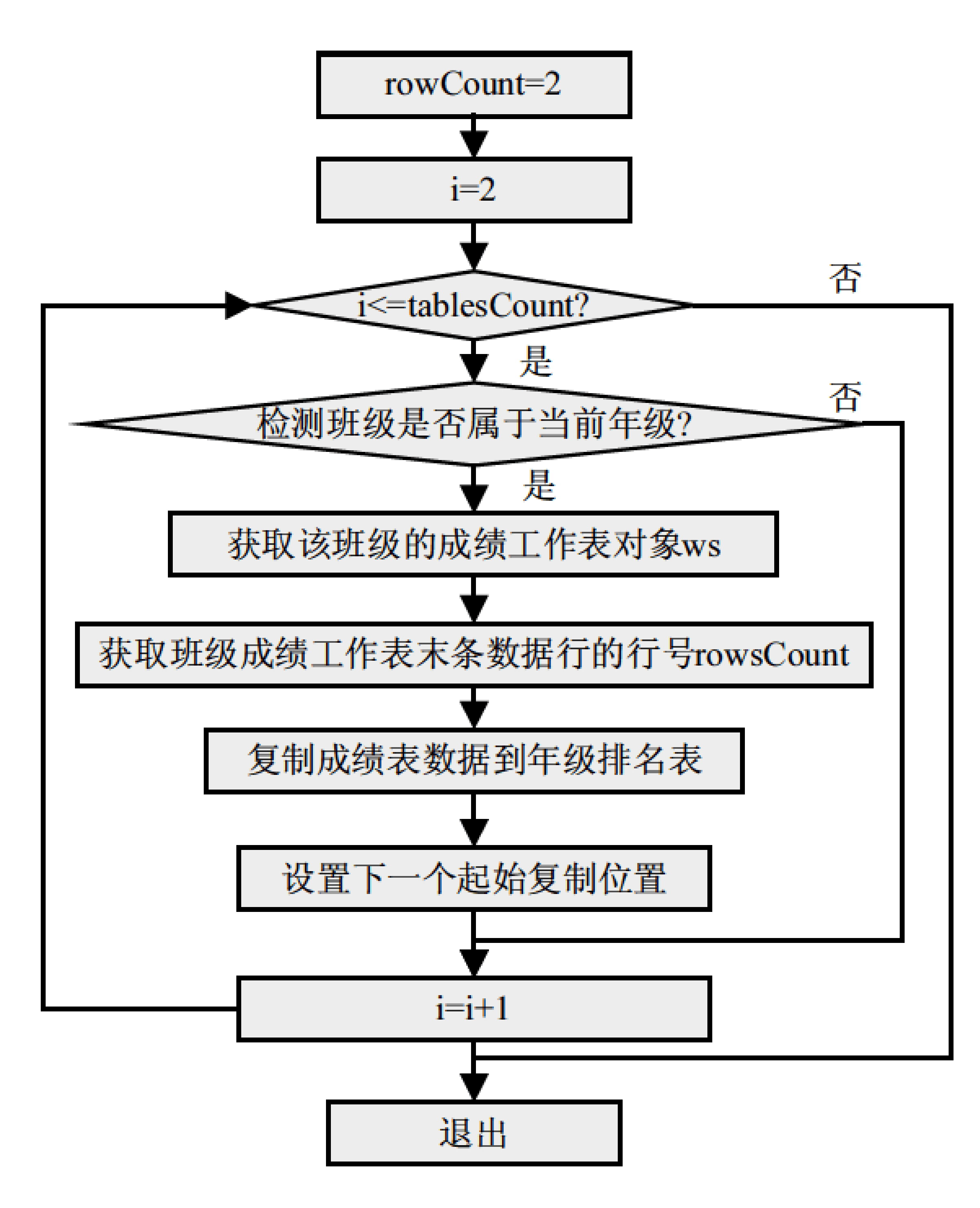


图 5-34　从各班级复制成绩数据到年级排名表流程图

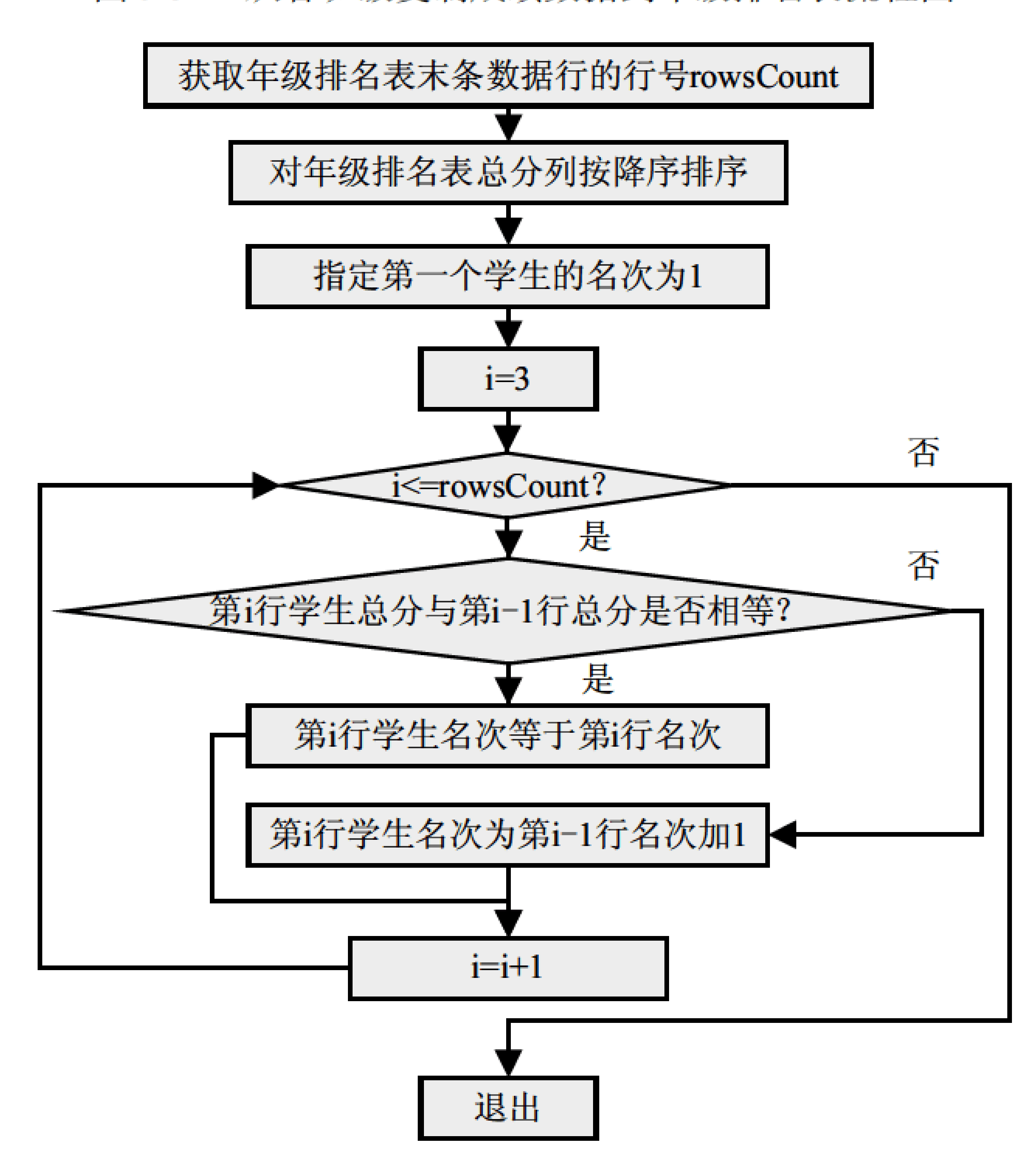


图 5-35　排序并设置名次流程图

以下是该宏过程的详细代码解释：

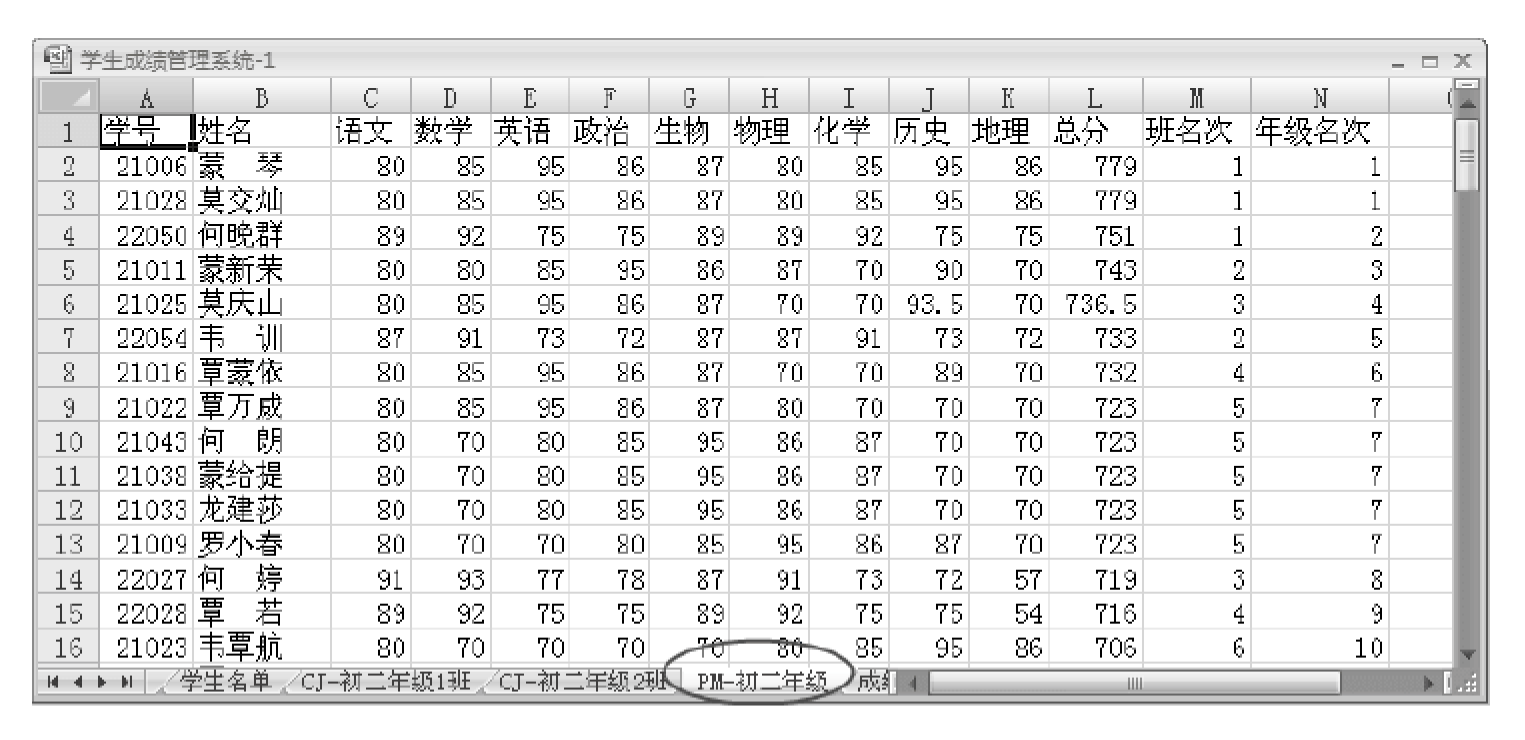
```
Sub NJPM()
Dim tablesCount As Integer
Dim rowsCount As Integer, i As Integer,
Dim strMsg As String
Dim totalBJ As String
Dim rowNumber As Integer
Dim ws As Worksheet
frmXSMD.Show
tablesCount = Sheet6.Cells(Rows.Count, 2).End(xlUp).Row      '获取年级班级表班级列末条的行号
For i = 2 To tablesCount                                     '循环年级班级表中所有班级列单元格
    If Left(Sheet6.Cells(i, 2), Len(tempNJ)) = tempNJ Then   '检测该班级是否属于当前年级
        On Error Resume Next
        Set ws = Sheets("CJ-" & Sheet6.Cells(i, 2))          '获取该班级的工作表对象
        If Err.Number Then
            MsgBox "班级：<" & Sheet6.Cells(i, 2) & ">的成绩表没有建立！ ", vbOKOnly
            Exit Sub
        End If
        Err.Clear                                            '清除所有错误记录
    End If
Next
Application.ScreenUpdating = False
rowNumber = 2                                                '复制数据起始行号
For i = 2 To tablesCount                                     '循环年级班级表中所有班级列单元格
    If Left(Sheet6.Cells(i, 2), Len(tempNJ)) = tempNJ Then   '检测该班级是否属于当前年级
        Set ws = Sheets("CJ-" & Sheet6.Cells(i, 2))              '获取该班级的成绩工作表对象
        rowsCount = ws.Cells(Rows.Count, 1).End(xlUp).Row        '获取班级成绩工作表末条数据行
                                                                  的行号
        ws.Range("A2:M" & rowsCount).Copy Sheet4.Range("A" & rowNumber)    '复制成绩数据
        rowNumber = rowNumber + rowsCount -1                 '设置下一次复制到年级排名表的位置
    End If
Next
Sheet4.Activate                                              '激活年级排名表
rowsCount = Sheet4.Cells(Rows.Count, 1).End(xlUp).Row        '获取年级排名表末条数据行的行号
Sheet4.Range("A1:N" & rowsCount).Sort Key1:=Range("L2"), Order1:=xlDescending, Header:= _
        xlGuess, OrderCustom:=1, MatchCase:=False, Orientation:=xlTopToBottom, _
        SortMethod:=xlPinYin, DataOption1:=xlSortNormal      '对总分列按降序排序
Sheet4.Cells(2, 14) = 1                                      '指定第一个学生的名次为 1
For i = 3 To rowsCount                                       '循环年级排名表所有学生记录行
    If Sheet4.Cells(i -1, 12) <> Sheet4.Cells(i, 12) Then    '检测 i 行学生总分与 i-1 行总分是否相等
        Sheet4.Cells(i, 14) = Sheet4.Cells(i -1, 14) + 1     '设置 i 行学生名次为 i-1 行名次加 1
    Else
        Sheet4.Cells(i, 14) = Sheet4.Cells(i -1, 14)         '设置 i 行学生名次等于 i 行名次
    End If
Next
Application.ScreenUpdating = True
End Sub
```

在年级排名表中包含了一个【保存排名表】按钮。单击该按钮时，将弹出一个对话框（如图 5-36 所示）。该对话框用于设置保存年级排名表的名称。当用户在【选择年级】复合框中

选择了相应年级后，程序将把年级排名表保存为以“PM-”开头，后接年级为名称的新工作表（如图 5-37 所示）。



图 5-36　保存年级排名表设置



学生成绩管理系统-1

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	学号	姓名	语文	数学	英语	政治	生物	物理	化学	历史	地理	总分	班名次	年级名次
2	21006	蒙　琴	80	85	95	86	87	80	85	95	86	779	1	1
3	21028	莫交灿	80	85	95	86	87	80	85	95	86	779	1	1
4	22050	何晚群	89	92	75	75	89	89	92	75	75	751	1	2
5	21011	蒙新茉	80	80	85	95	86	87	70	90	70	743	2	3
6	21025	莫庆山	80	85	95	86	87	70	70	93.5	70	736.5	3	4
7	22054	韦　训	87	91	73	72	87	87	91	73	72	733	2	5
8	21016	覃蒙依	80	85	95	86	87	70	70	89	70	732	4	6
9	21022	覃万成	80	85	95	86	87	80	70	70	70	723	5	7
10	21043	何　朗	80	70	80	85	95	86	87	70	70	723	5	7
11	21038	蒙给提	80	70	80	85	95	86	87	70	70	723	5	7
12	21033	龙建莎	80	70	80	85	95	86	87	70	70	723	5	7
13	21009	罗小春	80	70	70	80	85	95	86	87	70	723	5	7
14	22027	何　婷	91	93	77	78	87	91	73	72	57	719	3	8
15	22028	覃　若	89	92	75	75	89	92	75	75	54	716	4	9
16	21023	韦覃航	80	70	70	70	70	80	85	95	86	706	6	10

学生名单　CJ-初二年级1班　CJ-初二年级2班　PM-初二年级　成绩

图 5-37　保存年级排名

该按钮在执行宏时调用功能表模块中的 SaveTotalData 过程，该过程的流程图如图 5-38 所示。

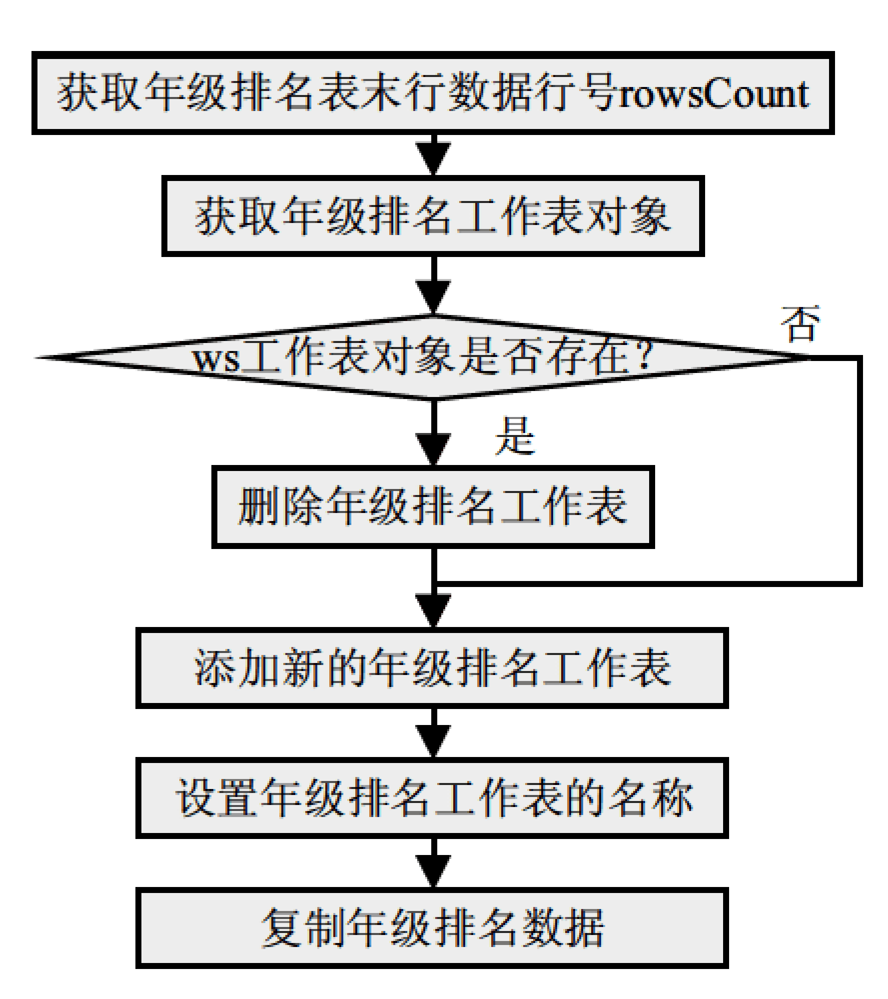


图 5-38　保存年级排名表流程图

以下是该过程的详细代码解释：

```
Sub SaveTotalData()
Dim ws As Worksheet, rowsCount As Integer
With frmXSMD
    .Caption = "选择年级名"                                  '设置窗体显示标签
    .Show                                                   '显示窗体
End With
rowsCount = Sheet4.Cells(Rows.Count, 1).End(xlUp).Row       '获取年级排名表末行数据行号
Application.ScreenUpdating = False
On Error Resume Next
Set ws = Sheets("PM-" & tempNJ)                             '获取年级排名工作表对象
If ws Is Nothing Then                                       '检测 ws 工作表对象是否存在
    Set ws = ThisWorkbook.Sheets.Add                        '添加新工作表
```

```
Else
    ws.Delete                                           '删除 ws 工作表
    Set ws = ThisWorkbook.Sheets.Add                    '添加新工作表
End If
ws.Name = "PM-" & tempNJ                                '设置年级排名工作表的名称
Sheet4.Range("A1:N" & rowsCount).Copy ws.Range("A1:N" & rowsCount)    '复制年级排名数据
ws.Columns.AutoFit                                      '年级排名表各列自动对齐
Application.ScreenUpdating = True
End Sub
```

5.4.3　成绩再处理模块设计

成绩再处理模块用于修改已保存好的班级成绩工作表。在已经保存的班级成绩工作表中，并没有重新计算总分、计算班级名次等功能按钮。为了能够在完成编辑班级成绩表的同时，完成重新计算总分以及名次等工作，需要将班级成绩工作表的数据导入到成绩输入表中完成班级成绩再处理工作。成绩再处理模块正是完成该部分工作。

在首页单击【成绩再处理】按钮后程序将执行菜单跳转模块中的 CJZCL 过程。程序首先以无模式方式显示【成绩再处理】对话框。该对话框用于获取需要处理的成绩表的名称。获取再处理成绩表名称后，程序激活了成绩输入表。激活该工作表后，程序清除了工作表所有数据，然后从再处理工作表中复制所有数据到成绩输入表中。该过程的代码不多，这里不再列出该过程的流程图。

以下是该过程的详细代码解释：

```
Sub CJZCL()
Dim rowsCount As Integer, ws As Worksheet
frmCJZCL.Show                                         '显示成绩再处理表选择窗口
Sheet3.Activate                                       '激活成绩输入表
Set ws = Sheets(tempTableName)                        '获取再处理工作表对象
rowsCount = ws.Cells(Rows.Count, 1).End(xlUp).Row     '获取成绩再处理工作表末条记录行号
Application.ScreenUpdating = False
Sheet3.Range("A2:M" & Rows.Count).ClearContents       '清除成绩输入表所有数据
ws.Range("A2:M" & rowsCount).Copy Sheet3.Range("A2")  '将再处理工作表数据复制到成绩输入表中
Application.ScreenUpdating = True
End Sub
```

5.5　查询模块设计

查询模块主要完成与学生成绩管理相关的查询工作。该模块可以分为班级学生查询、教师查询和班级成绩查询。查询工作的实现方法都采用了自动筛选的方式，而筛选条件通过自定义窗体获得。以下是这 3 个查询模块的功能介绍。

- 班级学生查询：该模块主要用于查询学生信息。用户在这里可以按年级、班级、学号、姓名查询学生情况。

- ❑ 教师查询：该模块可以查询某个班级的具体科目的任职教师。
- ❑ 班级成绩查询：该模块可以按班级、按学生号或按学生名查询学生成绩信息。

本节所涉及到的 3 个模块的代码不多，所有的功能几乎都是通过窗体实现的。关于窗体的设计请参见后面窗体设计章节的介绍，介绍时都是通过实例的形式加以说明。

5.5.1　班级学生查询设计

学生资料包括学号、年级、班级名与学生名等信息。这些信息都是预先在基本资料建立模块中已经建立的资料。当需要查询学生的相关资料时，可以通过学生查询模块完成该工作。该模块的操作流程如图 5-39 所示。

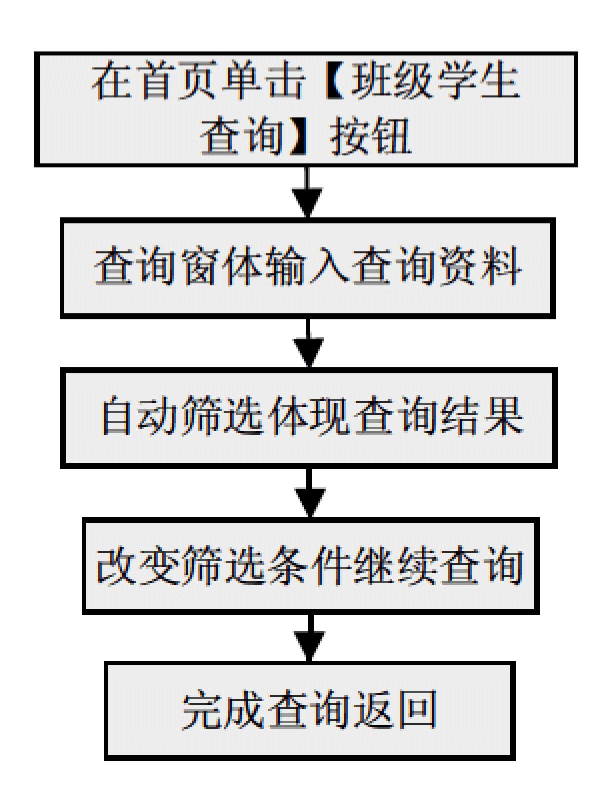


图 5-39　学生查询操作流程图

（1）在首页单击【查询】分类栏中的【班级学生查询】按钮。此时将执行“菜单跳转代码”模块中的 BJXSCX 过程。该过程将打开【学生查询】条件输入窗口，其详细代码如下：

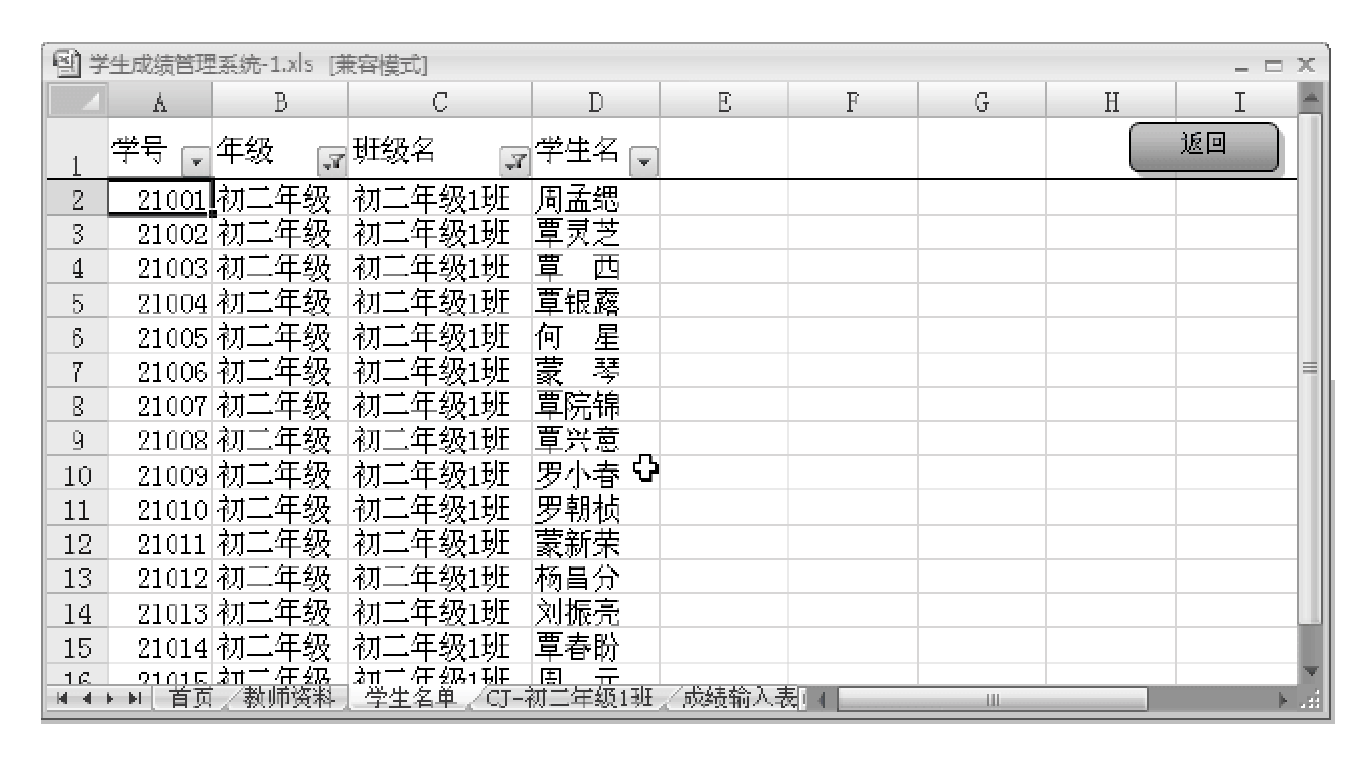
```
Sub BJXSCX()
    frmXSCX.Show
End Sub
```

（2）在弹出的【学生查询】条件设置窗口中输入学号、年级、班级和学生名查询信息。其中年级和班级信息可以通过下拉列表框获取，如图 5-40 所示。

（3）在【学生查询】条件设置窗口中输入完查询信息并单击【确定】按钮。此时将会跳转到“学生名单”工作表，在该表中将以自动筛选的形式显示查询结果。如果需要修改查询条件，可以单击自动筛选列首单元格右下方的下拉按钮并重新设置筛选条件，具体操作见本章知识点二。筛选后的效果如图 5-41 所示。



图 5-40　学生信息查询条件输入窗口



学号	年级	班级名	学生名
21001	初二年级	初二年级1班	周孟缌
21002	初二年级	初二年级1班	覃灵芝
21003	初二年级	初二年级1班	覃　西
21004	初二年级	初二年级1班	覃银霖
21005	初二年级	初二年级1班	何　星
21006	初二年级	初二年级1班	蒙　琴
21007	初二年级	初二年级1班	覃院锦
21008	初二年级	初二年级1班	覃兴意
21009	初二年级	初二年级1班	罗小春
21010	初二年级	初二年级1班	罗朝桢
21011	初二年级	初二年级1班	蒙新荣
21012	初二年级	初二年级1班	杨昌分
21013	初二年级	初二年级1班	刘振亮
21014	初二年级	初二年级1班	覃春盼

图 5-41　学生查询结果效果图

5.5.2　教师与科目查询设计

教师资料包括教师名、所教课程名以及对应班级名称等信息。这些信息也都是预先在基

本资料建立模块中已经建立的资料。当需要查询教师的相关资料时，可以通过教师查询模块完成该工作。操作流程如图 5-42 所示。

（1）在首页单击【查询】分类栏中的【教师查询】按钮。此时将执行“菜单跳转代码”模块中的 JSCX 过程。该过程将打开【教师查询】条件输入窗口。过程的详细代码如下：

```
Sub JSCX()
    frmJSCX.Show
End Sub
```

（2）在【教师查询】条件设置窗口中输入科目名和班级名查询信息。这两条查询信息都可以通过过下拉列表框获取，该窗口的界面如图 5-43 所示。

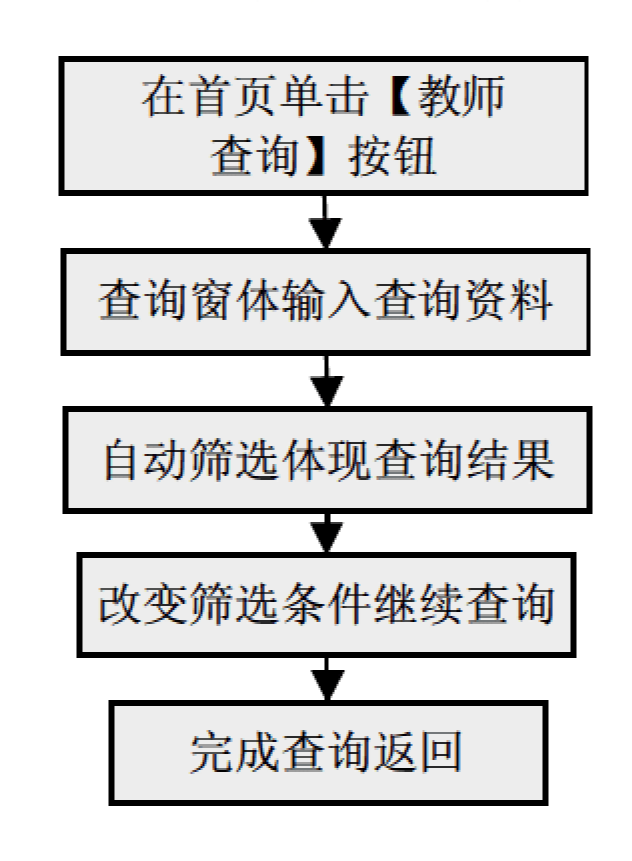


图 5-42　教师与科目查询流程图



图 5-43　教师信息查询条件输入窗口

（3）在【教师查询】条件设置窗口中输入查询信息并单击【确定】按钮。此时将会跳转到“教师资料”工作表并在该表中以自动筛选的方式显示查询结果。如果需要修改查询条件，可以单击自动筛选下拉按钮并重新设置筛选条件。筛选后的效果如图 5-44 所示。

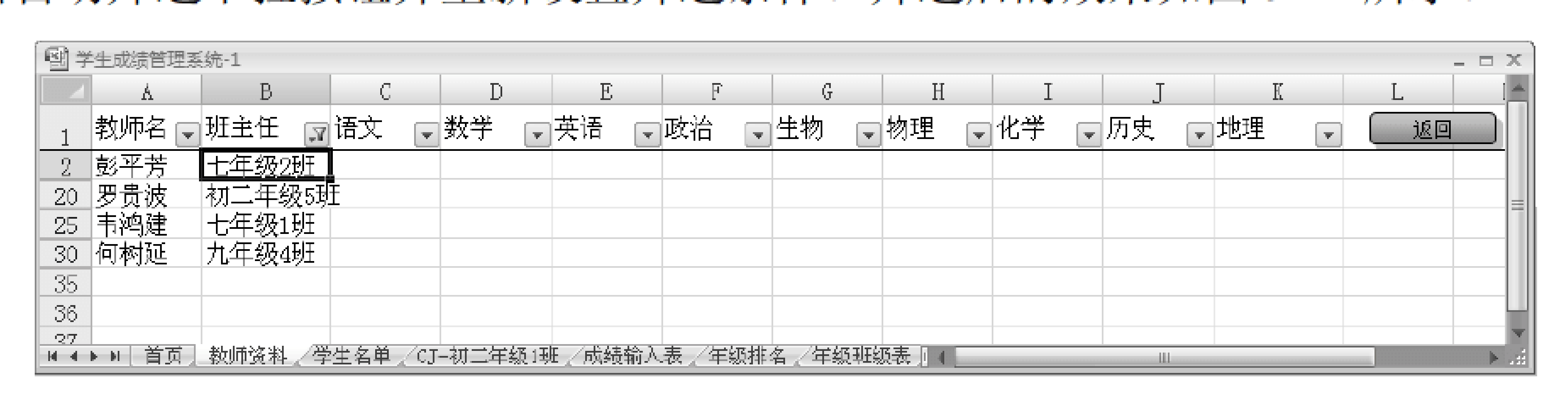


图 5-44　教师资料查询结果图

5.5.3　班级成绩查询设计

班级成绩查询设计模块可以快速定位某个班级或某个班级的某位学生的成绩信息。在查询班级或学生的成绩信息之前需要确保该班级的成绩表已经建立并且保存在工作簿中。当需要查询班级或班级中某位学生的成绩信息时，可以通过班级成绩查询模块完成该工作。该模块的操作过程如图 5-45 所示。

（1）在首页单击【查询】分类栏中的【班级成绩查询】按钮。此时将执行“菜单跳转代

码”模块中的 CJCX 过程。该过程将打开【成绩查询】条件输入窗口。该过程详细代码如下：

```
Sub CJCX()
frmCJCX.Show
End Sub
```

（2）在【成绩查询】条件设置窗口中输入班级名、学号和姓名查询信息。当用户需要查询某班级所有学生成绩时，只需要输入班级名即可。当需要查询某学生的成绩时，学号和姓名两个信息只需要输入其中之一，程序会自动寻找另一相应信息并将其输出到相应的文本框。该窗体界面如图 5-46 所示。

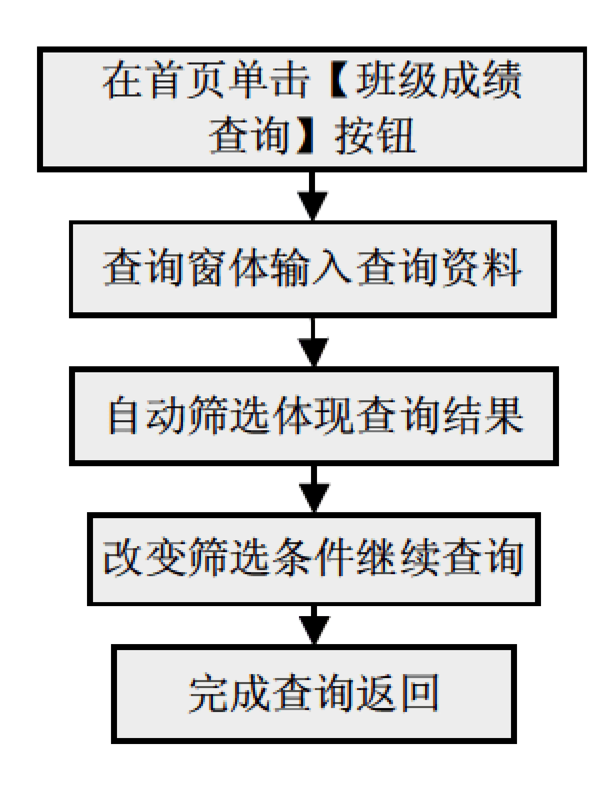


图 5-45　班级成绩查询流程图



图 5-46　班级成绩查询条件输入窗口

（3）在【成绩查询】窗口中输入查询信息并单击【确定】按钮。此时将会跳转到相应班级的成绩工作表中并以自动筛选的方式显示查询结果。如果需要修改查询条件，可以单击自动筛选下拉按钮并重新设置筛选条件。筛选后的效果如图 5-47 所示。

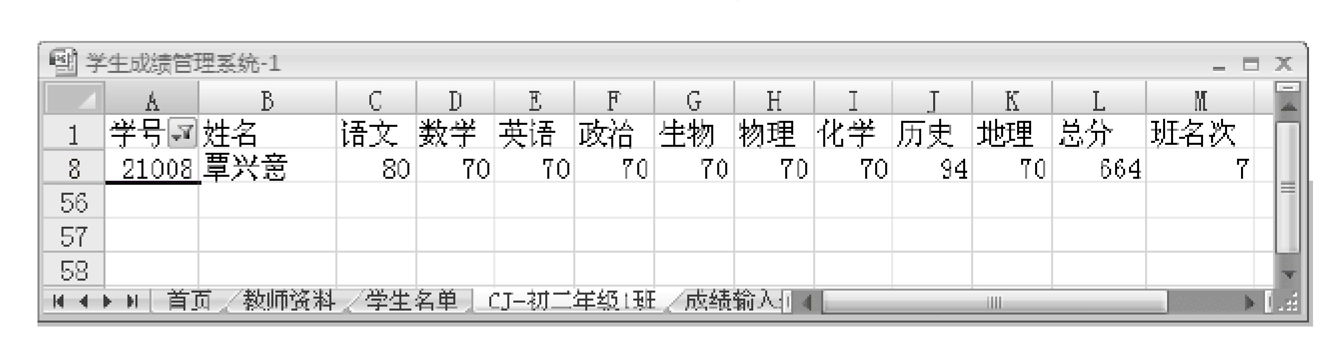


图 5-47　班级成绩查询结果

5.6　窗 体 设 计

在本实例中一共使用了 5 个窗体。这些窗体分别在各个工作表中被调用。部分窗体通过单击按钮激发，部分窗体通过双击工作表的单元格实现。本节将分别介绍这些窗体的界面设计过程与代码。

5.6.1　成绩查询设置窗口设计

当用户在首页中单击【班级成绩查询】按钮后，【成绩查询】对话框将会弹出。窗口中

包含了 3 个标签控件、1 个复合框控件、2 个文本框控件和 1 个按钮。如果用户需要查看整个班级的成绩情况，则在班级名复合框中选择相应班级即可。需要注意的是，当需要查询班级成绩时，必须已经完成班级成绩输入工作，否则班级名复合框没有任何项目可以选择。该窗口的界面如图 5-48 所示。

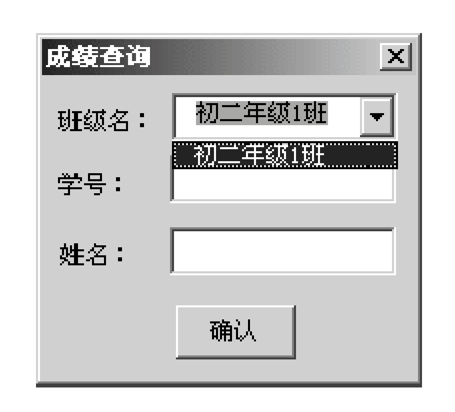


图 5- 48 【成绩查询】对话框

建立该窗口的步骤不算复杂，读者可以按照以下说明建立该窗口。

（1）在 Excel 2007 VBE 开发环境中依次选择【插入】|【用户窗体】命令。然后在属性窗口中修改新建立窗口的名称属性为 frmCJCX，如图 5-49 所示。

（2）在工具箱中选择标签控件。然后在 frmCJCX 窗体中连续建立 3 个标签控件。最后通过属性窗口将这 3 个标签的 Caption 属性分别修改为“班级名：”、“学号：”和“姓名：”。各个标签的相对位置如图 5-49 所示。

（3）在工具箱中选择复合框控件。然后在窗口的“班级名”标签右侧插入一个复合框控件。最后在属性窗口中将该复合框的名称属性修改为“cmb 班级名”，如图 5-50 所示。



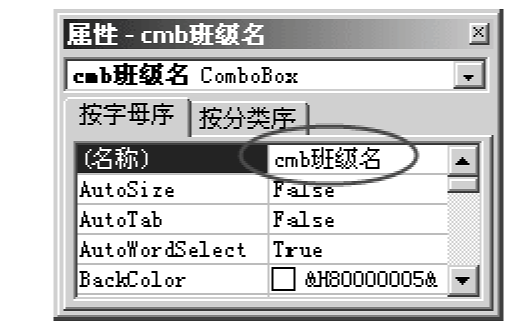


图 5-49 插入用户窗体操作示意图

图 5-50 班级名复合框名称设置

（4）在工具箱中选择文本框控件。然后在窗口的“学号”和“姓名”标签右侧各插入一文本框控件。最后在属性窗口中修改两文本框控件的名称属性分别为“txt 学号”和“txt 姓名”。

（5）在工具箱中选择按钮控件。然后在窗口的底部插入一按钮控件。随后在属性窗口中修改该按钮的 Caption 属性为“确认”，如图 5-49 所示。

成绩查询设置窗体包含了 5 个事件过程。这 5 个事件过程分别是窗口初始化事件、班级名复合框改变事件、姓名文本框改变事件、学号文本框改变事件和确定按钮单击事件。其中班级名改变事件只完成学号与姓名文本框显示状态的设置工作。这里不再介绍该事件的功能，其余 4 个事件的功能描述如下：

- 窗口初始化事件：窗口初始化时需要为班级名复合框添加项目。班级名复合框中的项目是工作簿中所有成绩工作表的名称。由于这些工作表的名称都带有“CJ-”的前缀，因而在添加项目时将这些前缀去除掉了。
- 学号文本框改变事件：学号文本框发生改变时，需要将当前学生号对应的学生名称

显示在姓名文本框中。程序使用 Find 方法定位该学生的学号单元格，然后通过 offset 方法获取该学号学生的姓名。

- 姓名文本框改变事件：姓名文本框发生改变时，需要将当前姓名学生的学号显示在学号文本框中。程序使用 Find 方法定位该学生的姓名单元格，然后通过 offset 方法获取该姓名学生的学号。
- 确定按钮单击事件：单击【确定】按钮时，程序首先检测用户是否输入了查询条件。在输入了必要的查询设置条件后，程序将激活成绩工作表。并且使用设置的查询条件，对成绩工作表进行筛选。

以下是这几个事件过程的详细代码解释：

```
Private Sub UserForm_Initialize()
Dim ws As Worksheet
With cmb 班级名
    .Clear                                          '清除班级名复合框所有项目
    For Each ws In ThisWorkbook.Worksheets          '循环工作簿中所有工作表
        If Left(ws.Name, 3) = "CJ-" Then            '检测工作表是否是成绩表
            .AddItem Right(ws.Name, Len(ws.Name) -3) '为班级名复合框添加新项目
        End If
    Next
End With
txt 学号.Enabled = False
txt 姓名.Enabled = False
Set ws = Nothing
End Sub

Private Sub cmb 班级名_Change()
txt 学号.Enabled = Len(cmb 班级名.Text)             '设置学号文本框的可用状态
txt 姓名.Enabled = Len(cmb 班级名.Text)             '设置姓名文本框的可用状态
End Sub

Private Sub CommandButton1_Click()
Dim ws As Worksheet
If Len(cmb 班级名.Text) Then                        '检测班级名复合框是否有输入结果
    Set ws = Worksheets("CJ-" & cmb 班级名.Text)    '获取成绩工作表对象
    ws.Activate                                     '激活成绩工作表
    If Len(txt 学号.Text) And Len(txt 姓名.Text) Then '检测用户是否输入学号和姓名
        ws.Range("A:A").AutoFilter field:=1, Criteria1:=txt 学号.Text   '筛选学号为用户选定学号的
                                                                        学生成绩行
    End If
End If
End Sub

Private Sub txt 姓名_Change()
Dim ws As Worksheet, rg As Range
Application.EnableEvents = False
Set ws = Worksheets("CJ-" & cmb 班级名.Text)        '获取成绩工作表对象
Set rg = ws.Range("B:B").Find(txt 姓名.Text)        '在工作表 B 列查找该名称的学生
```

```
If Not rg Is Nothing Then                          '检测是否找到该名称的学生单元格
    txt 学号.Text = rg.Offset(0, -1)               '设置学号文本框显示数据
Else
    txt 学号.Text = ""                             '清空学号文本框
End If
Application.EnableEvents = True
Set rg = Nothing
Set ws = Nothing
End Sub

Private Sub txt 学号_Change()
Dim ws As Worksheet, rg As Range
Application.EnableEvents = False
Set ws = Worksheets("CJ-" & cmb 班级名.Text)        '获取成绩工作表对象
Set rg = ws.Range("A:A").Find(txt 学号.Text)        '在工作表的 A 列查找该学号的学生
If Not rg Is Nothing Then                          '检测是否找到该学号的学生单元格
    txt 姓名.Text = rg.Offset(0, 1)                '设置姓名文本框显示数据
Else
    txt 姓名.Text = ""                             '清空姓名文本框
End If
Application.EnableEvents = True
Set rg = Nothing
Set ws = Nothing
End Sub
```

5.6.2 成绩再处理设置窗口设计

成绩再处理窗口用于选择需要重新处理的成绩工作表。在窗体中仅包含了 3 个控件，该窗体的界面十分简洁，制作也十分简单，这里不再列出制作该窗体的详细制作步骤。在复合框中包含了所有工作簿中已经建立了成绩资料的工作表的名称。【选择再处理成绩表】设置窗口。如图 5-51 所示。

图 5-51 【选择再处理成绩表】设置窗口

窗口中包含了两个事件过程：窗口初始化事件过程、确定按钮单击事件过程。这两个事件过程的详细代码解释如下：

```
Private Sub CommandButton1_Click()
tempTableName = "CJ-" & combTableName.Text         '保存需要重新处理成绩工作表的名称
Unload Me                                          '卸载窗口
End Sub

Private Sub UserForm_Initialize()
Dim ws As Worksheet
With combTableName
    .Clear                                         '清除复合框所有项目
    For Each ws In ThisWorkbook.Worksheets         '循环工作簿中所有工作表
        If Left(ws.Name, 3) = "CJ-" Then           '检测工作表是否是成绩工作表
            .AddItem Right(ws.Name, Len(ws.Name) -3)  '为复合框添加新项目
```

```
            End If
        Next
End With
End Sub
```

5.6.3　教师查询窗口设计

教师查询窗口用于设置查询教师所教授的科目以及所在班级。这两个查询设置将被应用到对教师资料工作表的查询操作中。该窗体的界面如图 5-52 所示，在该窗体中用户不能单独选择班级名。

在该窗体中包含了 3 个事件过程，分别是确定按钮单击事件、科目名复合框改变事件和窗口初始化事件。其中科目名复合框改变事件较为简单，只是用于设置科目名复合框的可用状态。这里不再对该过程的功能加以详细描述。以下是剩余两个事件过程的功能介绍。

- 窗口初始化事件：窗口被初始化时，需要完成两件事情。分别是为科目名复合框和班级名复合框添加项目、设置班级复合框的可用性。科目名复合框项目添加十分简单。而添加班级名复合框项目时，程序依次循环年级班级表中班级列所有单元格。最后将这些单元格的内容添加到班级名复合框中。
- 确定按钮单击事件：【确定】按钮用于将用户设置的筛选条件应用到教师资料工作表中。程序首先逐个比对用户设置的科目名称，根据该科目名称确认需要筛选的列号。最后对工作表进行筛选时，根据设置的班级名决定最终的筛选方式。如图 5-53 所示显示了该事件过程的执行流程。



图 5-52　教师查询设置窗口

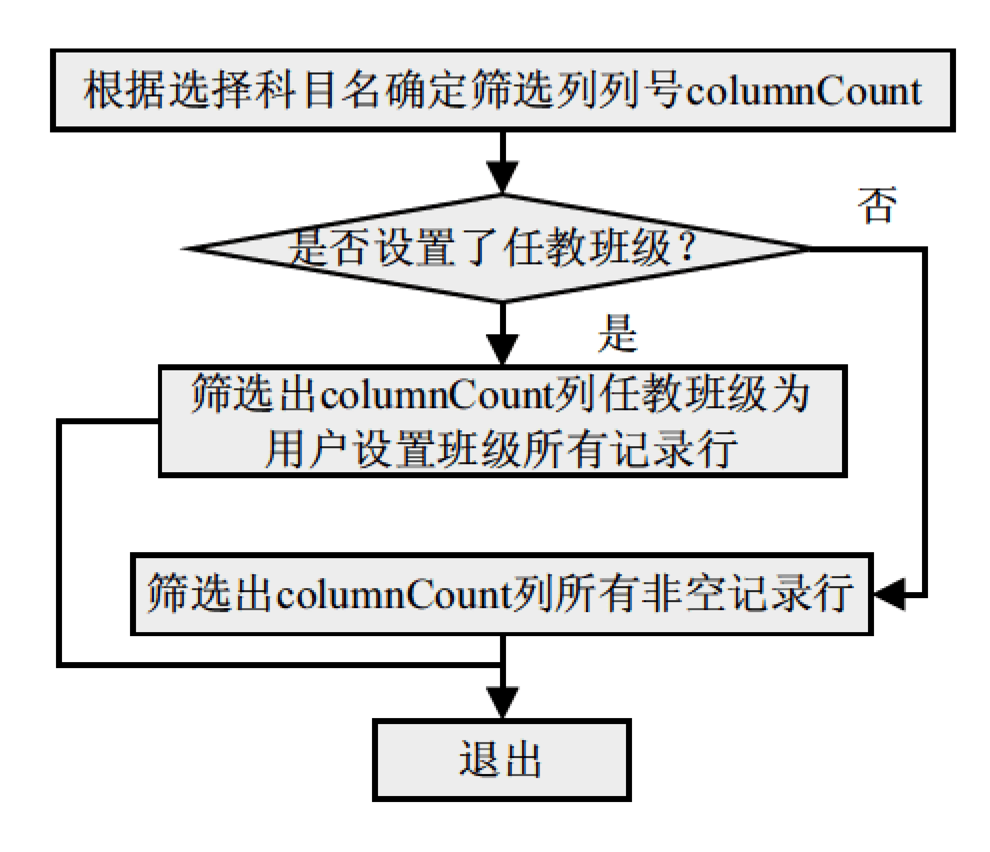


图 5-53　【确定】按钮单击事件过程流程图

该窗体的代码如下：

```
Private Sub UserForm_Initialize()
Dim rowsCount As Integer, i As Integer
With combKM
        .AddItem "班主任"                                    '添加第一个科目名项目
        .AddItem "语文"                                      '添加第二个科目名项目
        .AddItem "数学"                                      '添加第三个科目名项目
```

```
        .AddItem "英语"                                        '添加第四个科目名项目
        .AddItem "政治"                                        '添加第五个科目名项目
        .AddItem "生物"                                        '添加第六个科目名项目
        .AddItem "物理"                                        '添加第七个科目名项目
        .AddItem "化学"                                        '添加第八个科目名项目
        .AddItem "历史"                                        '添加第九个科目名项目
        .AddItem "地理"                                        '添加第十个科目名项目
End With
rowsCount = Sheet6.Cells(Rows.Count, 2).End(xlUp).Row          '获取年级班级工作表班级名列的末条
                                                                数据行号
For i = 2 To rowsCount                                         '循环班级名列所有班级
        combBJ.AddItem Sheet6.Cells(i, 2)                      '为班级名添加新项目
Next
combBJ.Enabled = False                                         '设置班级复合框不可用
End Sub

Private Sub btnOK_Click()
Dim columnCount As Integer
Sheet1.Activate                                                '激活教师资料表
Select Case combKM.Text
        Case Is = "班主任"                                     '检测科目名是否为班主任
                columnCount = 2                                '设置筛选列列号为 2
        Case Is = "语文"                                       '检测科目名是否为语文
                columnCount = 3                                '设置筛选列列号为 3
        Case Is = "数学"                                       '检测科目名是否为数学
                columnCount = 4                                '设置筛选列列号为 4
        Case Is = "英语"                                       '检测科目名是否为英语
                columnCount = 5                                '设置筛选列列号为 5
        Case Is = "政治"                                       '检测科目名是否为政治
                columnCount = 6                                '设置筛选列列号为 6
        Case Is = "生物"                                       '检测科目名是否为生物
                columnCount = 7                                '设置筛选列列号为 7
        Case Is = "物理"                                       '检测科目名是否为物理
                columnCount = 8                                '设置筛选列列号为 8
        Case Is = "化学"                                       '检测科目名是否为化学
                columnCount = 9                                '设置筛选列列号为 9
        Case Is = "历史"                                       '检测科目名是否为历史
                columnCount = 10                               '设置筛选列列号为 10
        Case Is = "地理"                                       '检测科目名是否为地理
                columnCount = 11                               '设置筛选列列号为 11
End Select
Application.ScreenUpdating = False
With Columns("A:K")
        .AutoFilter                                            '开启自动筛选
        If Len(combBJ.Text) Then                               '检测用户是否设置了筛选班级
                '筛选出 columnCount 列中任教班级为 combBJ.text 的所有教师信息
                .AutoFilter Field:=columnCount, Criteria1:="=*" & combBJ.Text & "*", Operator:=xlAnd
        Else
                '筛选出 columnCount 列中不为空的所有记录行
```

```
            .AutoFilter Field:=columnCount, Criteria1:="<>"
        End If
End With
Application.ScreenUpdating = True
End Sub

Private Sub combKM_Change()
combBJ.Enabled = Len(combKM.Text)
End Sub
```

5.6.4　学生信息查询窗口设计

学生信息查询窗口中要设置查询学生信息的条件。窗口中可以设置的查询条件包含学号、年级名、班级名以及学生名。用户查询信息时，可以同时设置一个或多个查询条件。该窗口一共包含了 4 个标签控件、2 个文本框控件和 2 个复合框控件，其界面如图 5-54 所示。

建立该窗口的步骤如下：

（1）在 Excel 2007 VBE 开发环境中依次选择【插入】|【用户窗体】命令。随后在属性窗口中修改该窗体的名称属性为 frmXSCX，同时也修改其 Caption 属性为“学生查询”，如图 5-55 所示。

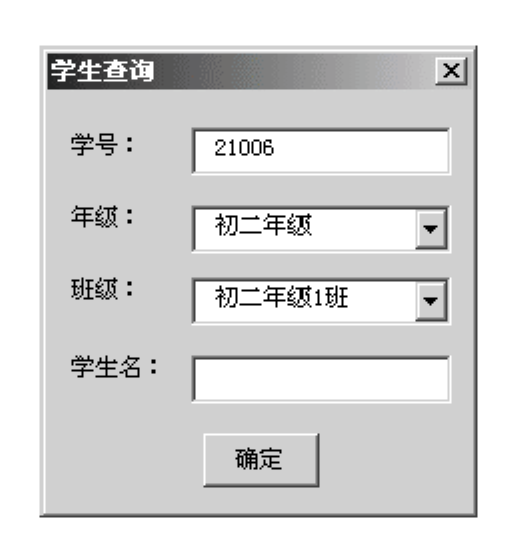


图 5-54　学生信息查询设置窗口

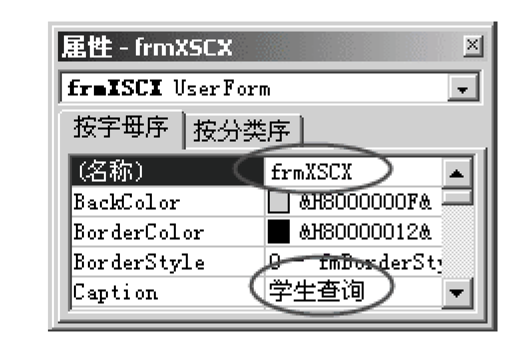


图 5-55　学生信息查询窗体属性设计

（2）在工具箱中选择标签控件。然后在窗体中连续建立 4 个标签控件。最后将这 4 个标签控件的 Caption 属性分别修改为“学号：”、“年级：”、“班级：”和“学生名”。最终各个标签的实际效果如图 5-56 所示。

（3）在工具箱中选择文本框控件。然后在“学号”和“学生名”标签右侧各插入一个文本框控件。最后修改这两个文本框的名称属性分别为 txtXH 和 txtXSM。

（4）在工具箱中选择复合框控件。然后在窗体的“年级”和“班级”标签右侧各插入一个复合框控件。最后在属性窗口中将这两个复合框控件的名称属性分别修改为 combNJ 和 combBJ。

（5）在工具箱中选择按钮控件。然后在窗体的底部插入一个按钮控件。然后在属性窗口中修改该按钮的 Caption 属性为“确定”。

在窗口中输入了合适的查询条件并单击【确定】按钮后，程序将自动获取这些查询设置并按照该设置进行查询。图 5-54 设置的查询条件是初二年级 1 班的 21006 号学生信息，该项查询设置条件下所获得的查询结果如图 5-57 所示。

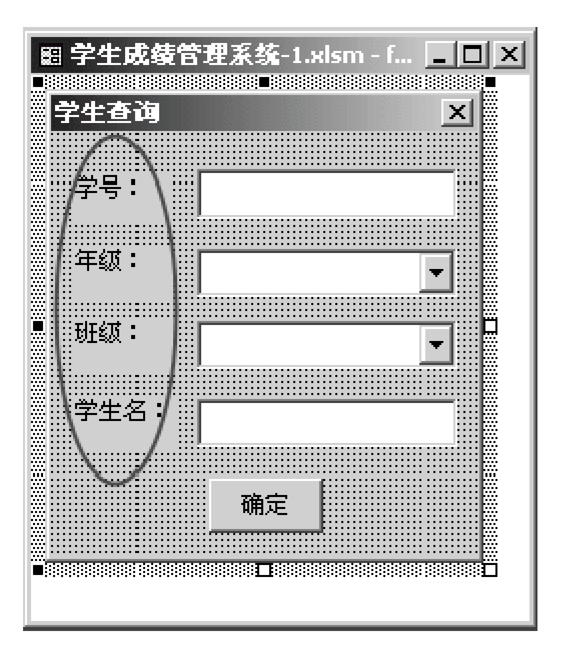


图 5-56　标签设计效果示意图



图 5-57　学生信息查询结果

在该窗口中一共包含了 3 个事件过程。这 3 个事件过程分别是窗口初始化事件过程、年级复合框改变事件过程和确定按钮单击事件过程。其中只有年级复合框改变事件过程的流程稍微复杂，下面的介绍中只给出该过程的流程图。这 3 个过程的功能介绍如下：

- 窗口初始化事件过程：窗口初始化时，需要初始化一些公共变量。这些变量将用于保存用户在设置窗口中选择查询条件。另外该过程还需要完成窗口中年级与班级复合框项目的添加工作。
- 年级复合框改变事件过程：当用户改变年级复合框中的选择项目时，程序需要根据新的年级信息确认班级复合框中应该显示的可选项目。首先程序将班级复合框中所有项目清除，然后程序从年级班级工作表的班级名列中获取当前选择年级的所有班级，最后将这些班级依次作为新项目添加到班级复合框中。该过程的流程图如图 5-58 所示。

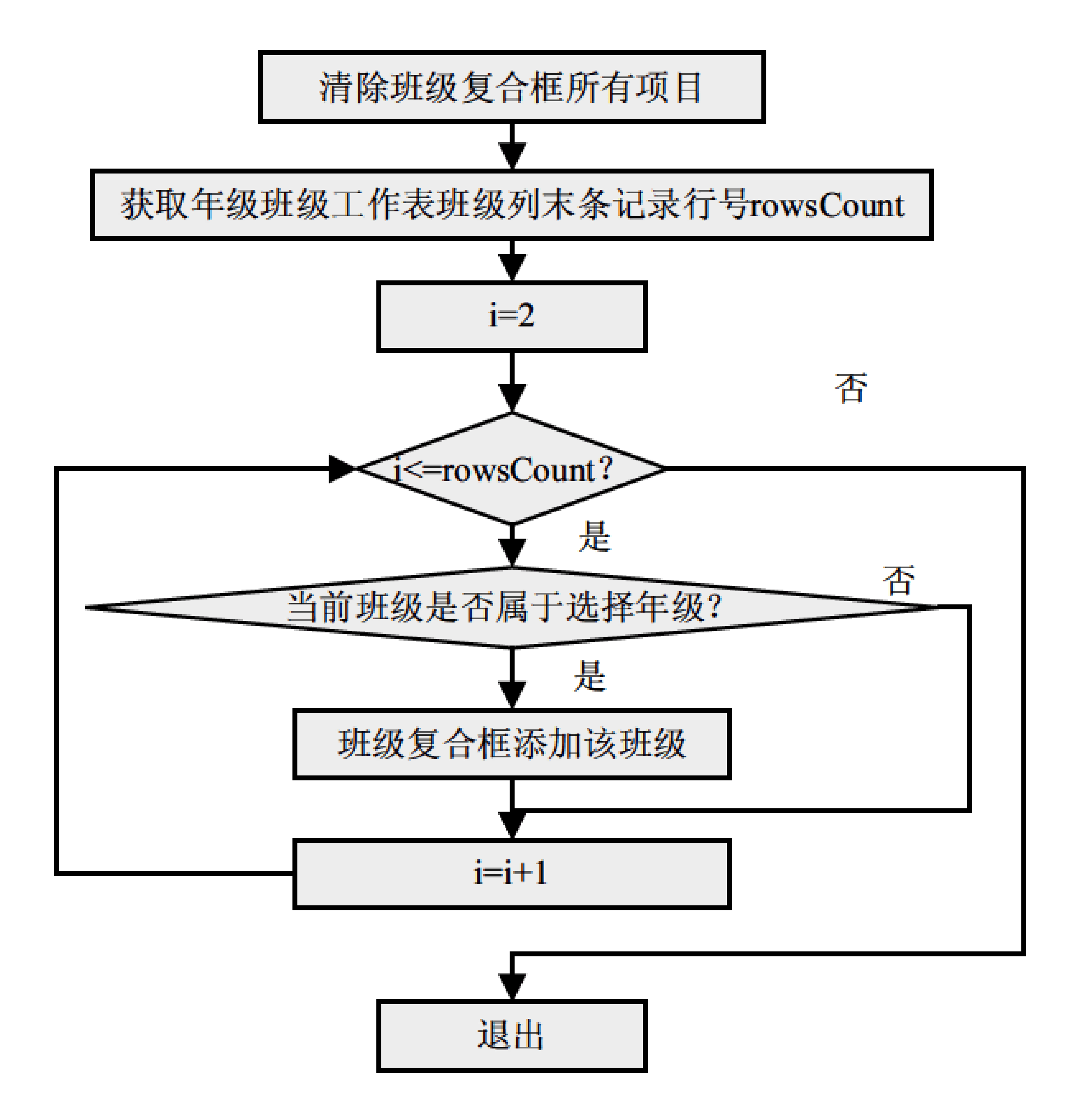


图 5-58　年级复合框改变事件过程流程图

- 确定按钮单击事件过程：单击【确定】按钮时，程序将把用户选择的筛选设置条件

保存到公共变量中。然后根据这些条件分别完成不同的筛选工作。

以下是该窗体 3 个事件过程的详细代码解释：

```
'窗口初始化代码
Private Sub UserForm_Initialize()
Dim rowsCount As Integer, i As Integer
'将临时存储查询条件的各个变量置空
tempXH = ""                                                 '置空学号公共变量
tempNJ = ""                                                 '置空年级公共变量
tempBJ = ""                                                 '置空班级公共变量
tempXSM = ""                                                '置空学生名公共变量
'获取年级列表，并将其添加到年级下拉列表框中
With combNJ
    .Clear                                                  '清除年级复合框中所有项目
    rowsCount = Sheet6.Cells(Rows.Count, 1).End(xlUp).Row   '获取年级班级表中年级名列末条
                                                             数据行行号
    For i = 2 To rowsCount                                  '循环年级名列所有数据单元格
        combNJ.AddItem Sheet6.Cells(i, 1)                   '为年级名复合框添加新项目
    Next
End With
'获取所有班级资料列表，并将其列入班级下拉列表框中
With combBJ
    .Clear                                                  '清除班级名复合框中所有项目
    rowsCount = Sheet6.Cells(Rows.Count, 2).End(xlUp).Row   '获取年级班级表中班级名列末条
                                                             数据行行号
    For i = 2 To rowsCount                                  '循环班级名列所有数据单元格
        combBJ.AddItem Sheet6.Cells(i, 2)                   '为班级名复合框添加新项目
    Next
End With
End Sub

'年级下拉列表框改变事件
'该事件用于检测年级输入是否改变，当发生改变时，修改班级下拉列表框内容
Private Sub combNJ_Change()
Dim rowsCount As Integer, i As Integer
With combBJ
    .Clear                                          '清空班级下拉列表框项目
    '据班级年纪表计算班级列行数
    rowsCount = Sheet6.Cells(Rows.Count, 2).End(xlUp).Row
    For i = 2 To rowsCount                          '循环检测班级年纪表的班级列
        '当班级的名称对应了当前的年级时，将该班级添加到班级下拉列表框的项目中
        If combNJ.Text = Left(Sheet6.Cells(i, 2), Len(combNJ.Text)) Then
            combBJ.AddItem Sheet6.Cells(i, 2)       '为班级复合框添加新项目
        End If
    Next
End With
End Sub

'按钮确定单击事件过程
```

```
Private Sub btnOK_Click()
'将查询条件保存到相应的变量中
tempXH = txtXH.Text                                    '保存学号查询条件
tempNJ = combNJ.Text                                   '保存年级查询条件
tempBJ = combBJ.Text                                   '保存班级查询条件
tempXSM = txtXSM.Text                                  '保存学生名查询条件
Sheet2.Activate                                        '激活学生名单表
Application.ScreenUpdating = False                     '关闭屏幕刷新
'对学生名单表的 A 到 D 列实施自动筛选
With Columns("A:D")
    .AutoFilter                                        '开启自动筛选
    If Len(tempXH) Then                                '检测用户是否输入了学号
        .AutoFilter Field:=1, Criteria1:=tempXH        '对学号列实施自动筛选
    End If
    If Len(tempNJ) Then                                '检测用户是否输入了年级
        .AutoFilter Field:=2, Criteria1:=tempNJ        '对年级列实施自动筛选
    End If
    If Len(tempBJ) Then                                '检测用户是否输入了班级
        .AutoFilter Field:=3, Criteria1:=tempBJ        '对班级列实施自动筛选
    End If
    If Len(tempXSM) Then                               '检测用户是否输入了学生名
        '对学生名实施自动筛选
        .AutoFilter Field:=4, Criteria1:="=*" & tempXSM & "*", Operator:=xlAnd
    End If
End With
Application.ScreenUpdating = True                      '重新打开屏幕刷新
End Sub
```

5.6.5 年级班级选择窗口设计

在本实例中有很多功能都需要选择年级与班级，程序通过一个年级班级选择窗口实现了该功能。打开该窗口时，会随着完成事务的不同，窗口显示的标题也会有所差别。该窗口的界面如图 5-59 所示。在该窗口中共包含了两个标签控件、两个复合框控件以及两个按钮控件。



图 5-59 年级班级获取窗口

建立该窗口步骤比较简单，以下简述该窗口的建立过程：

（1）在 Excel 2007 VBE 开发环境中依次选择【插入】|【用户窗体】命令，随后在属性窗口中修改该窗体的名称属性为 frmXSMD，如图 5-60 所示。

（2）在工具箱中选择标签控件。随后在窗体中连续建立两个标签控件。然后在属性窗口中将这两个标签的 Caption 属性分别修改为“选择年级：”和“选择班级：”。标签设计效果示意图如图 5-61 所示。

（3）在工具箱中选择复合框控件。随后在“选择年级”和“选择班级”标签右侧各插入一个复合框控件。然后在属性窗口中将这两复合框控件的名称属性分别修改为 combNJ 和 combBJ。

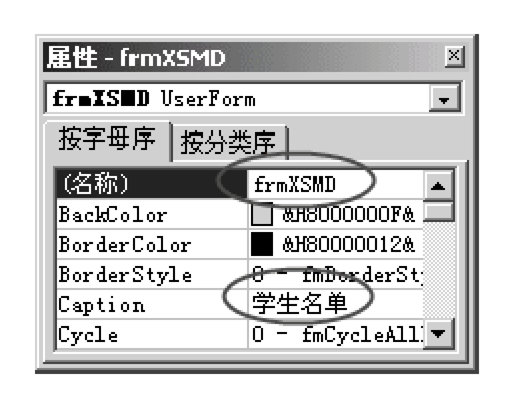


图 5-60　学生名单窗体属性设计

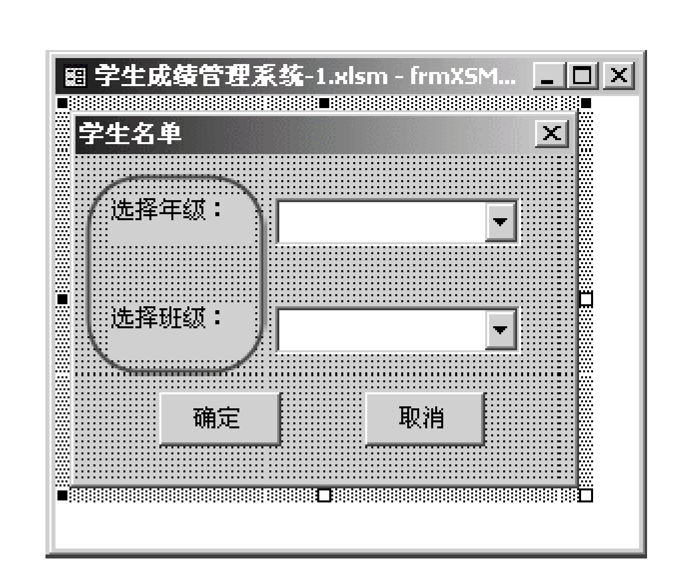


图 5-61　标签设计效果示意图

（4）在工具箱中选择按钮控件。随后在窗体的底部连续插入两个按钮控件。然后在属性窗口中修改这两个按钮控件的 Caption 属性分别为“确定”和“取消”。

在年级班级选择窗口中一共包含了 4 个事件过程代码。这 4 个事件分别为：窗口初始化事件、年级复合框改变事件、确定按钮单击事件和取消按钮单击事件。其中取消按钮仅仅完成退出窗口的工作，下面只对前 3 个事件的功能加以详细描述。

- 窗口初始化事件：当窗口第一次加载时，需要初始化年级复合框的项目以及初始化部分公共变量。复合框项目的数据来源是年级班级工作表的年级名列。程序首先获取该列最后一条数据的行号，然后遍历该列的各个单元格，将这些单元格的数据作为新项目添加到年级复合框中。图 5-62 是该事件过程的流程图。
- 年级复合框改变事件：当用户在窗口的年级复合框中选择了某个年级或更改了该项设置时，会激发年级复合框改变事件。该事件用于激活班级复合框的可用状态并为该复合框添加选择项目。程序首先将班级复合框设置为可用，然后对年级班级工作表的班级名列逐行检测。当检测单元格的班级是当前年级时，将该班级名作为新项目添加到班级复合框中。如图 5-63 所示是该过程的流程图。

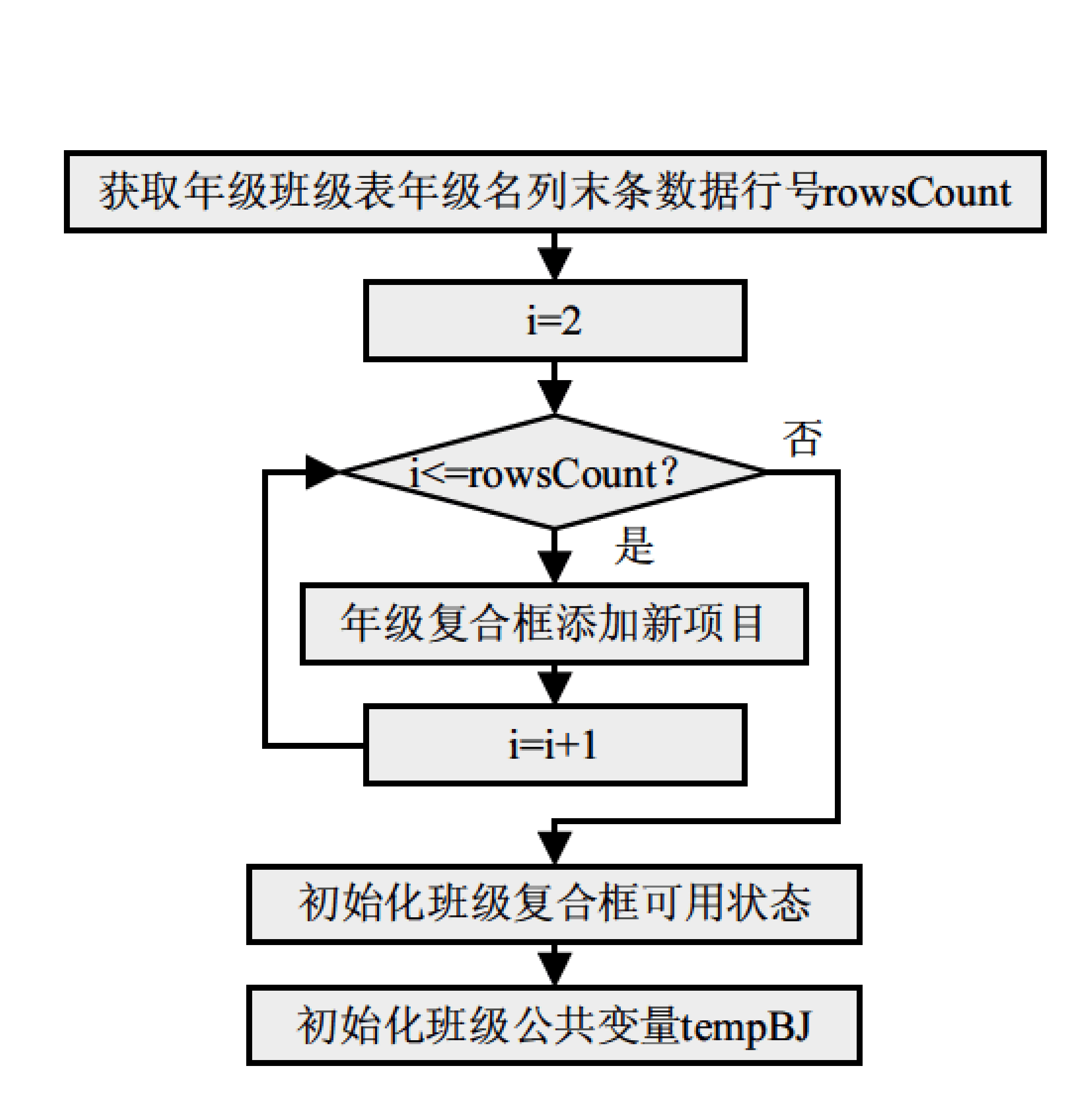


图 5-62　年级班级窗口初始化过程流程图

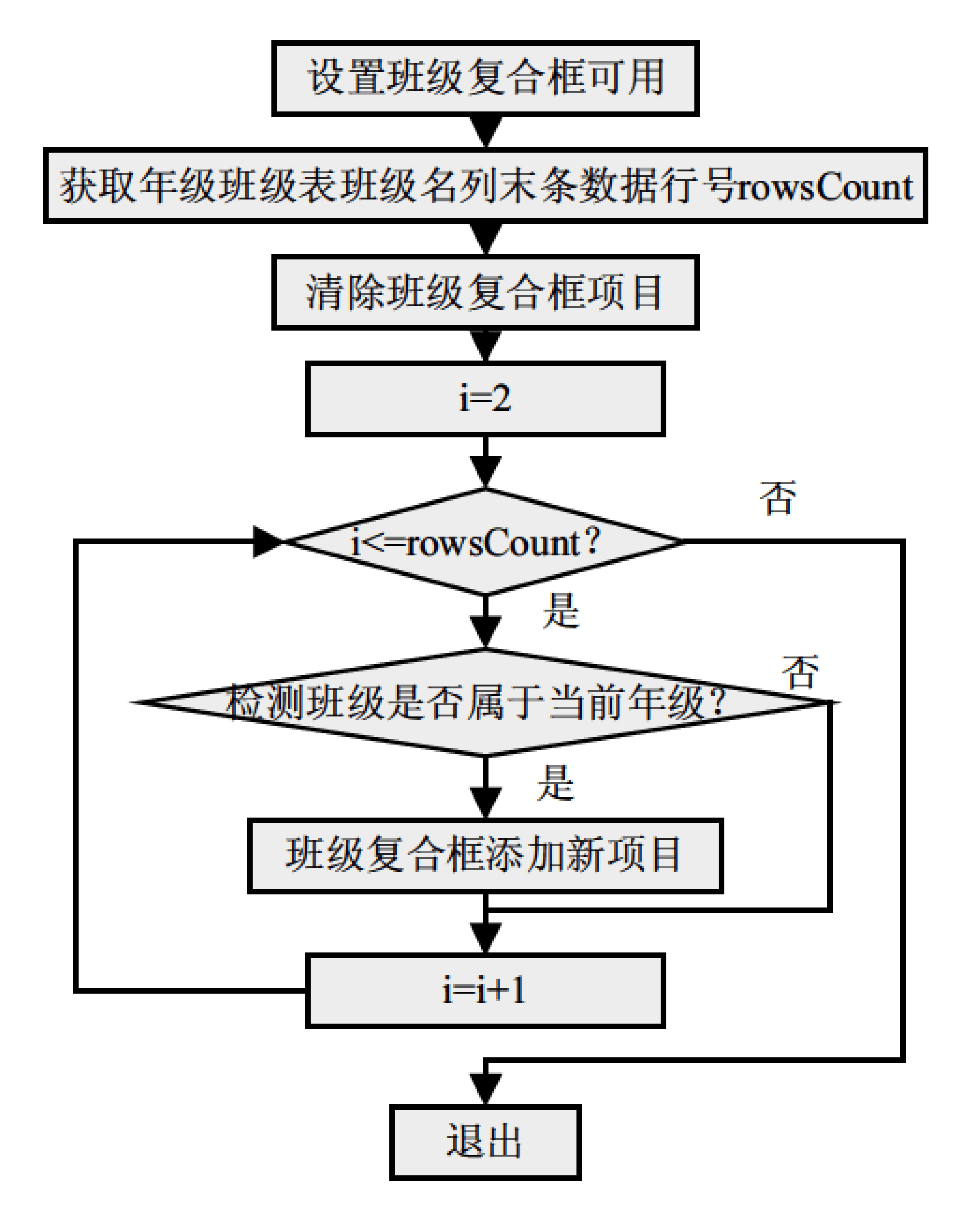


图 5-63　年级复合框改变事件流程图

- 确定按钮单击事件：单击【确定】按钮时，程序将把用户所做的选择设置保存到公共变量中。该事件的代码比较简单，这里不再列出该过程的流程图。

以下是年级班级选择窗口的详细代码解释：

```
Private Sub UserForm_Initialize()                              '窗口初始化
Dim rowsCount As Integer                                       '保存表的最大行数
Dim i As Integer                                               '循环计数变量
Application.ScreenUpdating = False                             '关闭屏幕刷新，以防屏幕闪动
rowsCount = Sheet6.Cells(Rows.Count, 1).End(xlUp).Row
'将年级下拉列表框按照年级班级表中的年级列填充
For i = 2 To rowsCount
    combNJ.AddItem Sheet6.Cells(i, 1)
Next
combBJ.Enabled = False                                         '禁止班级下拉列表框的可用性
Application.ScreenUpdating = True                              '回复屏幕刷新
tempBJ = ""                                                    '将公共变量置空
End Sub

Private Sub combNJ_Change()                                    '当年级下拉列表框被修改后执行该代码
Dim rowsCount As Integer                                       '保存表的最大行数
Dim i As Integer                                               '循环计数变量
combBJ.Enabled = True                                          '回复班级下拉列表框的可用性
'计算年级班级表中班级列最后有数据的单元格行数
rowsCount = Sheet6.Cells(Rows.Count, 2).End(xlUp).Row
'将属于当前年级设置下的班级名称序列写入班级下拉列表框的序列中
With combBJ
    .Clear
    For i = 2 To rowsCount
        If Left(Sheet6.Cells(i, 2), Len(combNJ.Text)) = combNJ.Text Then
            .AddItem Right(Sheet6.Cells(i, 2), _
Len(Sheet6.Cells(i, 2)) -Len(combNJ.Text))
        End If
    Next
End With
End Sub

Private Sub btnOK_Click()
'如果在班级下拉列表框选择班级，把班级信息（包括年级）保存在公共变量 tempBJ 中
'负责仅保存年级到公共变量 tempNJ
If Len(combBJ.Text) Then
    tempBJ =Trim(combNJ.Text ) & Trim(combBJ.Text)
Else
    tempNJ =Trim(combNJ.Text)
End If
Unload Me                                                      '卸载窗口
End Sub

Private Sub CommandButton1_Click()
Unload Me
End Sub
```

5.7　系 统 测 试

本部分系统测试以生成一个年级学生成绩排名为目的。测试的内容主要是班级成绩表建立与年级排名表生成的功能。该项测试假设初二年级只有两个班级，并且学生名单工作表中已经建立了两个班级学生的信息。以下将分两个小节分别测试。

5.7.1　建立班级成绩

要产生年级成绩排名，首先需要建立该年级下所有班级的成绩。系统根据年级班级表中的信息来确认某个年级所包含的班级数目，因而在年级班级表中务必输入完整的设置信息。以下是建立初二年级 1 班成绩的步骤：

（1）在首页单击【成绩输入】按钮，在随后弹出的对话框中设置年级与班级名称分别为“初二年级”、“1 班”，然后单击【确定】按钮，如图 5-64 所示。

（2）设置了年级与班级名称后，弹出成绩输入表工作表。在该表中输入所有学生的所有科目成绩即可，如图 5-65 所示。

图 5-64　设置年级与班级信息

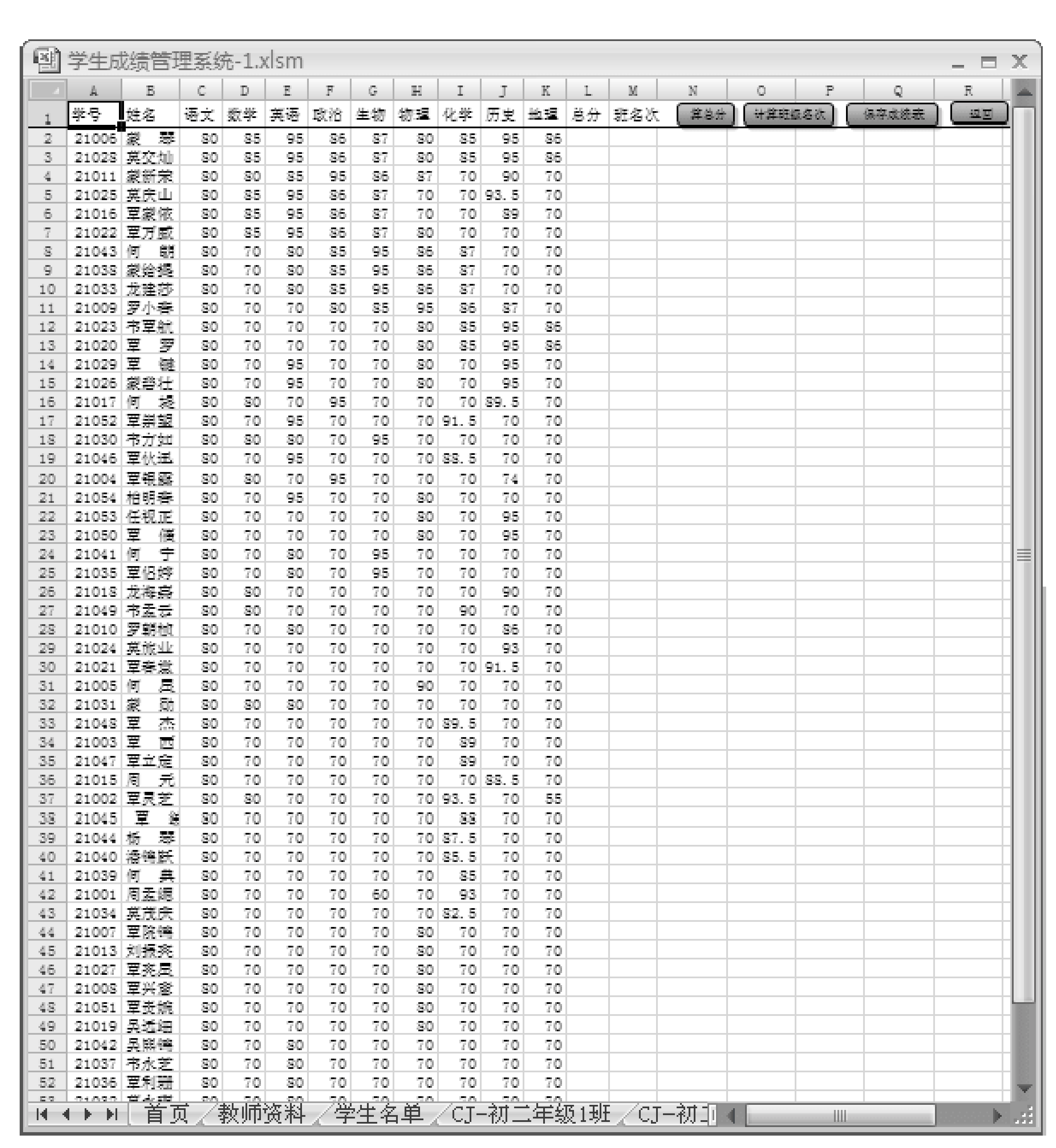


图 5-65　初二年级 1 班学生成绩

（3）输入完成绩后，在该工作表中依次单击【算总分】与【计算班级名次】按钮，此时表中的总分和班名次栏将自动计算获取数据，如图 5-66 所示。

（4）计算总分和班级名次完成后，单击【保存成绩表】按钮。在随后弹出的对话框中设

置保存的成绩表所属年级和班级，最后单击窗口中的【确定】按钮，如图 5-67 所示。

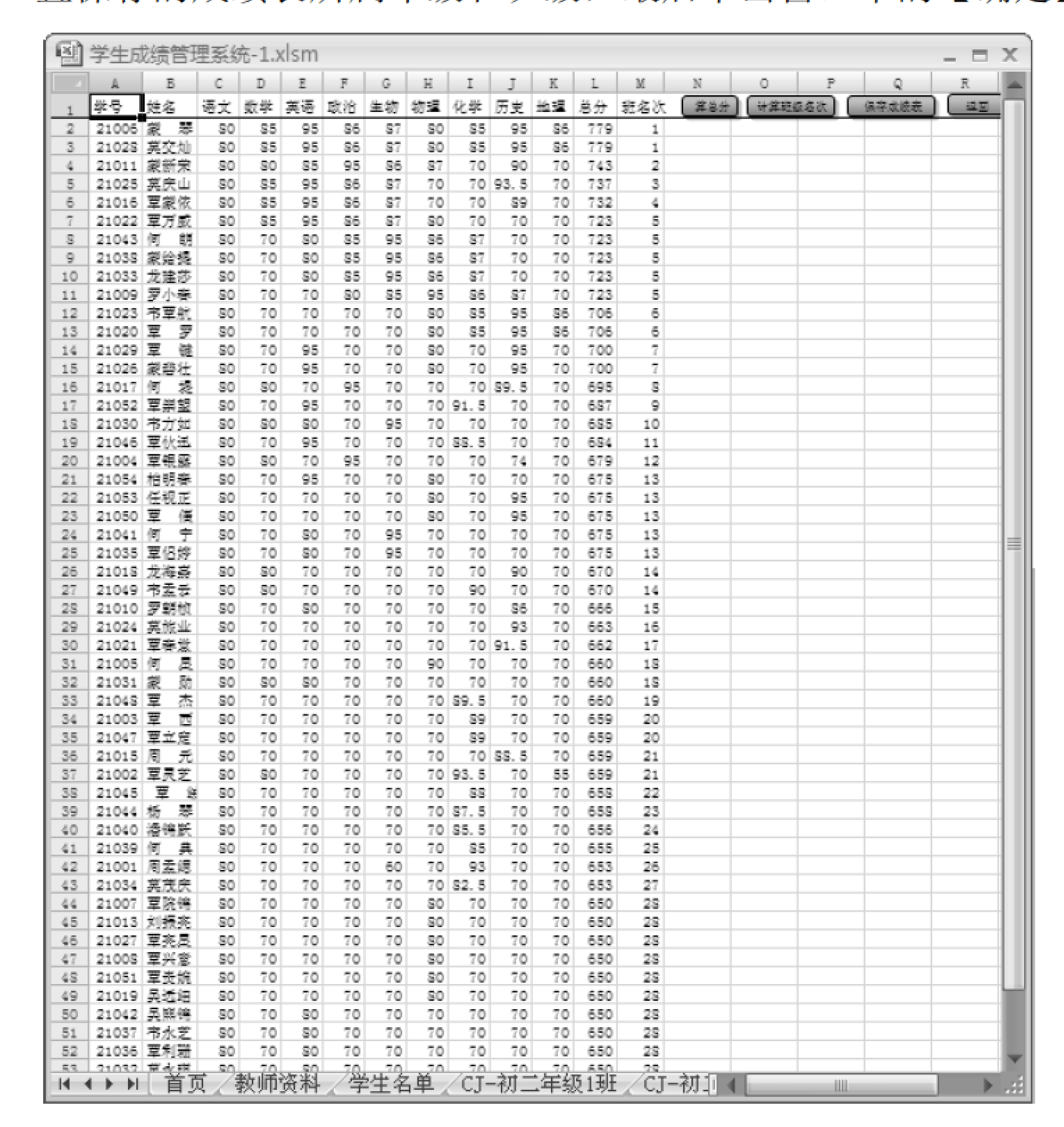


图 5-66　计算总分和班级名次



图 5-67　设置保存成绩表所属年级与班级

5.7.2　生成年级成绩排名

由于初二年级有两个班级，在生成年级排名前，务必建立另外一个班级的成绩表。其步骤与上面建立初二年级 1 班的成绩表类似，这里不再加以说明。以下是建立年级排名表的步骤：

（1）在首页单击【年级排名】按钮，在随后弹出的对话框中，设置年级为“初二年级”，单击【确定】按钮，如图 5-68 所示。

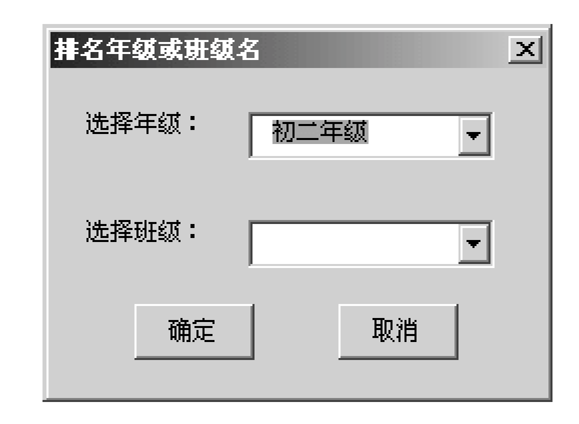
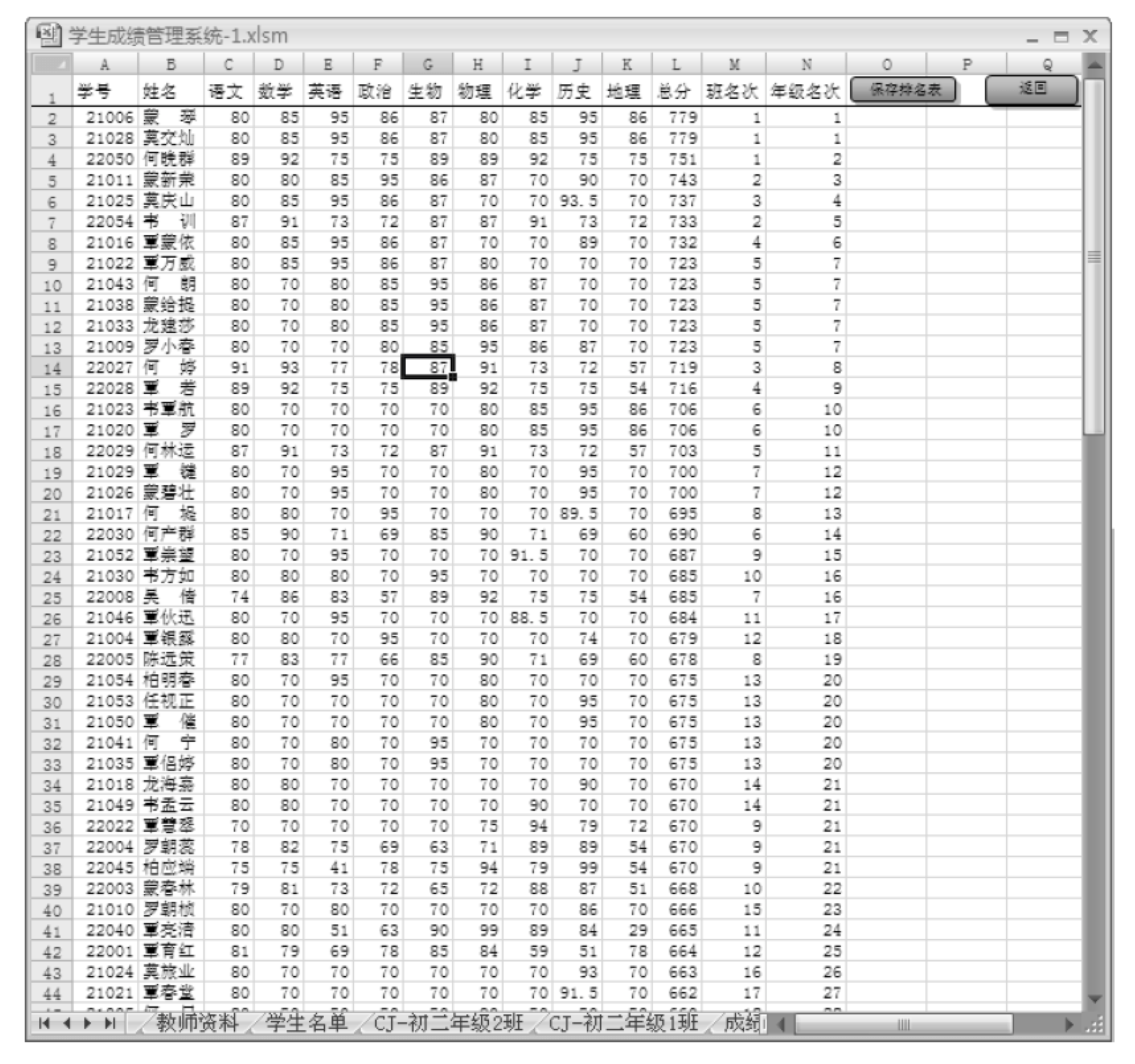


图 5-68　设置排名年级所属年级

（2）在上一步设置了年级并确认后，程序自动计算该年级所有班级学生的名次，如图 5-69 所示。当需要保存该名次表时，只需要单击【保存排名表】按钮，在随后弹出的窗口中设置



排名所属年级即可，如图 5-70 所示。



	学号	姓名	语文	数学	英语	政治	生物	物理	化学	历史	地理	总分	班名次	年级名次
2	21006	蒙　琴	80	85	95	86	87	80	85	95	86	779	1	1
3	21028	莫文灿	80	85	95	86	87	80	85	95	86	779	1	1
4	22050	何晓群	89	92	75	75	89	89	92	75	75	751	1	2
5	21011	蒙新荣	80	80	85	95	86	87	70	90	70	743	2	3
6	21025	莫庆山	80	85	95	86	87	70	70	93.5	70	737	3	4
7	22054	韦　训	87	91	73	72	87	87	91	73	72	733	2	5
8	21016	覃蒙依	80	85	95	86	87	70	70	89	70	732	4	6
9	21022	覃万威	80	85	95	86	87	80	70	70	70	723	5	7
10	21043	何　朗	80	70	80	85	95	86	87	70	70	723	5	7
11	21038	蒙给挺	80	70	80	85	95	86	87	70	70	723	5	7
12	21033	龙建莎	80	70	80	85	95	86	87	70	70	723	5	7
13	21009	罗小春	80	70	70	80	85	95	86	87	70	723	5	7
14	22027	何　婷	91	93	77	78	87	91	73	72	57	719	3	8
15	22028	覃　岩	89	92	75	75	89	92	75	75	54	716	4	9
16	21023	韦覃航	80	70	70	70	70	80	85	95	86	706	6	10
17	21020	覃　罗	80	70	70	70	70	80	85	95	86	706	6	10
18	22029	何林运	87	91	73	72	87	91	73	72	57	703	5	11
19	21029	覃　键	80	70	95	70	70	80	70	95	70	700	7	12
20	21026	蒙碧壮	80	70	95	70	70	80	70	95	70	700	7	12
21	21017	何　姮	80	80	70	95	70	70	70	89.5	70	695	8	13
22	22030	何产群	85	90	71	69	85	90	71	69	60	690	6	14
23	21052	覃崇望	80	70	95	70	70	70	91.5	70	70	687	9	15
24	21030	韦方如	80	80	80	70	95	70	70	70	70	685	10	16
25	22008	吴　倩	74	86	83	57	89	92	75	75	54	685	7	16
26	21046	覃伙迅	80	70	95	70	70	70	88.5	70	70	684	11	17
27	21004	覃银露	80	80	70	95	70	70	70	74	70	679	12	18
28	22005	陈远贵	77	83	77	66	85	90	71	69	60	678	8	19
29	21054	柏明春	80	70	95	70	70	80	70	70	70	675	13	20
30	21053	任视正	80	70	70	70	70	80	70	95	70	675	13	20
31	21050	覃　催	80	70	70	70	70	80	70	95	70	675	13	20
32	21041	何　宁	80	70	80	70	95	70	70	70	70	675	13	20
33	21035	覃侣婷	80	70	80	70	95	70	70	70	70	675	13	20
34	21018	龙海嘉	80	80	70	70	70	70	70	90	70	670	14	21
35	21049	韦孟云	80	80	70	70	70	70	90	70	70	670	14	21
36	22022	覃慧琴	70	70	70	70	70	75	94	79	72	670	9	21
37	22004	罗朝蕊	78	82	75	69	63	71	89	89	54	670	9	21
38	22045	柏应端	75	75	41	78	75	94	79	99	54	670	9	21
39	22003	蒙春林	79	81	73	72	65	72	88	87	51	668	10	22
40	21010	罗朝桢	80	70	80	70	70	70	70	86	70	666	15	23
41	22040	覃亮涪	80	80	51	63	90	99	89	84	29	665	11	24
42	22001	覃育红	81	79	69	78	85	84	59	51	78	664	12	25
43	21024	莫旅业	80	70	70	70	70	70	70	93	70	663	16	26
44	21021	覃春堂	80	70	70	70	70	70	70	91.5	70	662	17	27

图 5-69　年级成绩排名表



图 5-70　设置排名表保存信息

第 6 章　固定资产管理系统

固定资产指使用期限较长，单位价值较高，并且在使用过程中保持原有实物形态的资产。固定资产管理系统是一个企事业单位不可缺少的部分。它的内容对于企事业单位的决策者和管理者来说都至关重要。

根据现行行业制度规定，企业使用了一年以上的房屋、建筑物、机器、设备、运输工具等资产，均应作为固定资产。不属于生产经营主要设备的物品，单位价值在 2000 元以上，并且使用期限超过两年的，也应作为固定资产。不符合上述条件的劳动资料，企业应作为低值易耗品管理和核算。

6.1　系 统 概 述

固定资产是企业的重要生产资料之一。随着时间的推移，固定资产会被逐渐损耗，其价值会逐渐减少。如果不对固定资产进行折旧管理，将会虚增企业资产，同时也虚增了企业利润，更严重的是由此也导致企业上交的所得税的增加。从这里可以看出固定资产的管理的重要性。实现固定资产的电算化可以极大地减少固定资产管理工作的工作量，也保证了该项工作的正确率。

6.1.1　设计思路

本系统针对功能需求，把系统功能划分为 4 个部分：系统基本设置、固定资产登记、固定资产折旧和固定资产统计分析。该系统的详细结构图如图 6-1 所示。

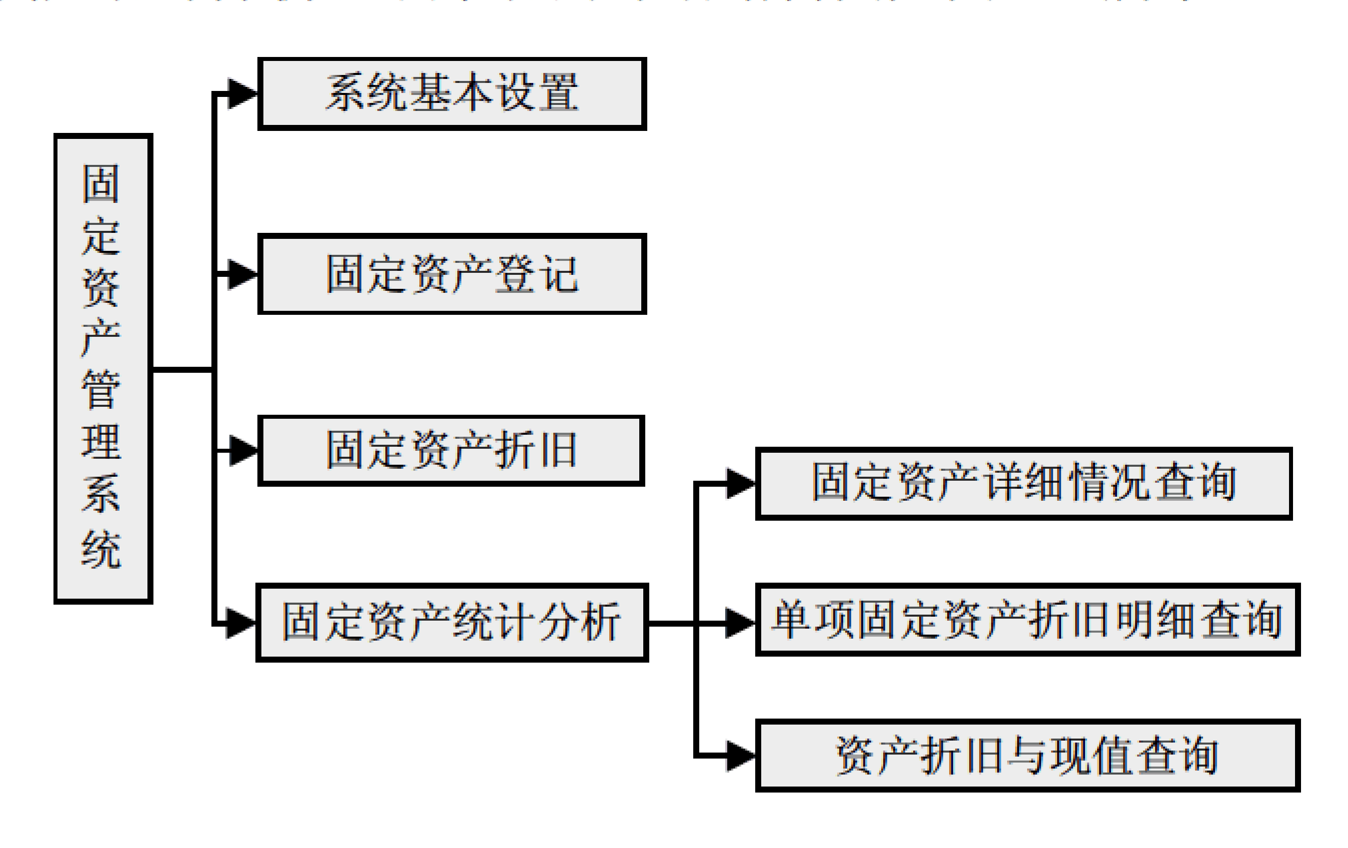


图 6-1　固定资产管理系统功能结构图

以下为该系统 4 个功能模块的详细描述。

- 系统基本设置。该功能块主要针对系统进行初始设置。这些设置包括资产编号、使用部门、资产类别、资产来源。这些信息都存储在对应的设置表中。
- 固定资产登记。该功能块主要用于新增固定资产项目，并且设置该固定资产的所有固定属性，其中有些属性是必填项目。
- 固定资产折旧。该功能块主要针对各个固定资产项目，生成固定资产折旧。折旧的计算结果都保存在资产项目的明细表中。
- 固定资产统计分析。该功能块主要用于统计各个固定资产的信息，以便于查询。该部分包括所有固定资产详细情况查询、单项固定资产折旧明细查询、资产折旧与现值查询 3 个部分。

固定资产进行折旧时，方法很多，例如平均年限法、工作量法、双倍余额递减法、年数总和法等。本例计算折旧的方法是平均年限法，其计算公式如下：

年折旧率=(1−净残值率)/使用年限

月折旧率=年折旧率/12

月折旧额=固定资产原值×月折旧率

该系统共包含了 6 个工作表，这些表按是否有功能代码可以分为无代码基础表和有代码功能表两类。代码基础表包括首页、单项固定资产折旧明细模板、设置表。有代码功能表包括固定资产登记表、固定资产登记统计、固定资产折旧与现值统计。各表的详细功能解释如下：

- 首页：用来显示系统所有功能和快速在各个功能间跳转。
- 单项固定资产折旧明细模板：该表是单项固定资产折旧明细表的格式范本。单项固定资产折旧明细表是通过复制该表然后向其中填入需要信息完成的。
- 设置表：存储系统的一些基本设置信息。
- 固定资产登记表：该表完成固定资产登记与保存工作。
- 固定资产登记统计：该表显示所有已登记的固定资产的详细信息列表。在其中双击某个固定资产行时，会跳转到该固定资产的详细折旧明细表。
- 固定资产折旧与现值统计：实时根据所有已有的固定资产详细折旧明细表产生固定资产折旧与现值报告。

6.1.2　知识点一：设置单元格条件格式

条件格式有助于突出显示所关注的单元格或单元格区域、强调异常值，还可以使用数据条、色阶和图标集来直观地显示数据。条件格式基于用户条件设置自动更改单元格区域的外观。如果条件为真（True），则该单元格的条件格式设置就会生效；如果条件为假（False），则该单元格的条件设置将不发生作用，单元格的格式恢复到原始状态。手动新建条件格式设置的方式如下：

选择表中的一个或多个单元格，在【开始】选项卡上的【样式】组中单击【条件格式】



下边的箭头（如图 6-2 所示），选择【新建规则】命令，弹出【新建格式规则】窗口（如图 6-3 所示），在该窗口中设置条件以及对应格式即可。



图 6-2　条件格式按钮组

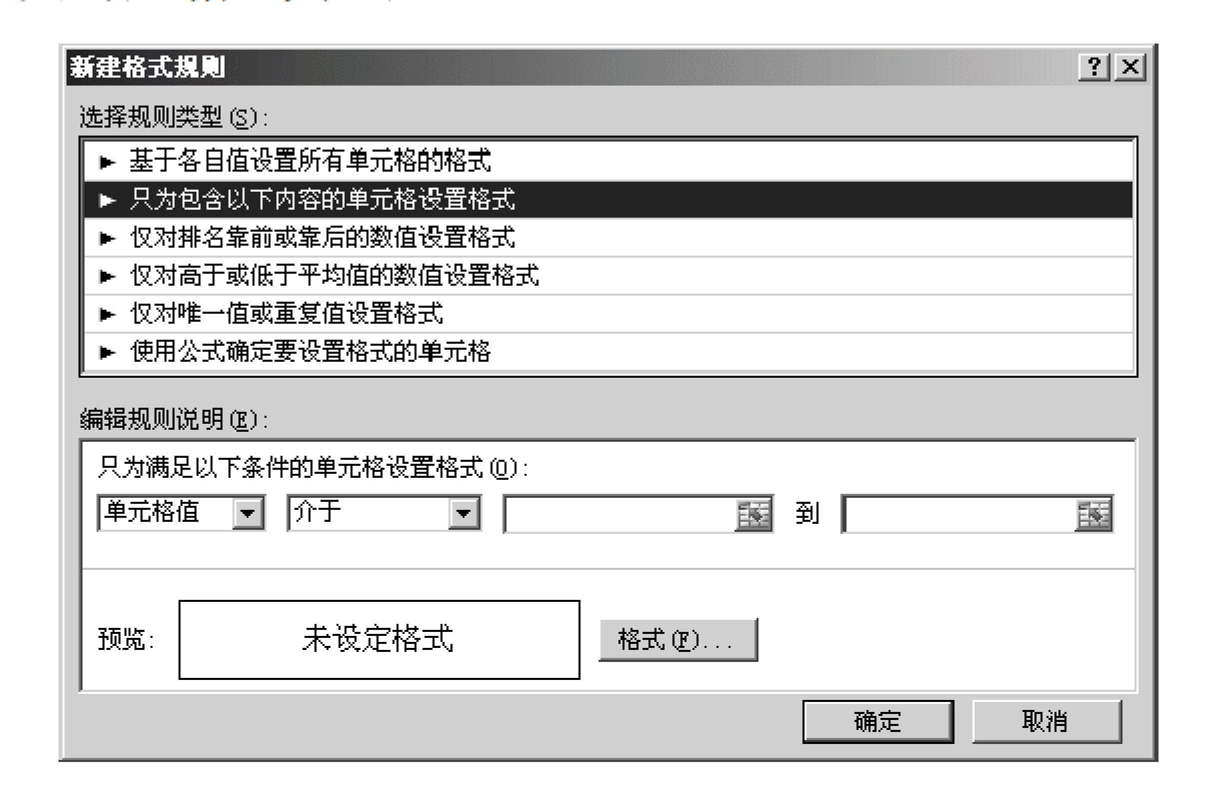


图 6-3　新建格式规则

6.1.3　知识点二：SendKey 方法

SendKey 方法用于将按键信息发送给应用程序，通过该方法可以模拟按键。该方法的使用语法如下：

```
Application.SendKeys(Keys, Wait)
```

其中 Keys 参数为必选，该参数以文本形式发送击键或组合键。Wait 参数为可选，如果为 True，则程序等到处理完按键后返回给宏；如果为 False（或者省略该参数），则继续运行宏而不等处理完按键。以下的实例表示按 Esc 键：

```
Application.SendKeys "{ESC}"
```

6.2　首页界面设计

首页工作表属于无代码基础表，该工作表的设计稍微复杂，所以单独列出一节加以讲述。首页被设计用来显示系统所有功能块以及快速在各个功能间跳转。在本系统的首页上可以通过对应按钮直接访问系统的各个功能。这些按钮包括固定资产登记、计提折旧、基本设置、查看资产登记表、查看单项资产和资产折旧与现值。在各个子功能上都设计有相应的返回按钮。这些按钮共同组成完整的跳转模式。

6.2.1　首页组成元素

本系统首页构成元素比较简单，共包括了 9 个形状和 3 个横向文本框。其中 9 个形状中又分 1 个矩形形状、2 个对角圆角矩形和 6 个棱台形状。各个形状功能如下：

- 矩形形状。首页界面外边框，界定首页界面的范围。

- 对角圆角矩形。划分功能区域，在对应的该形状上有相应的文本框显示该区域的功能。
- 棱台形状。被设计为跳转按钮。
- 文本框。包括 3 个文本框，1 个是首页标题，其他 2 个起功能提示作用。

系统首页的最终效果图如图 6-4 所示。

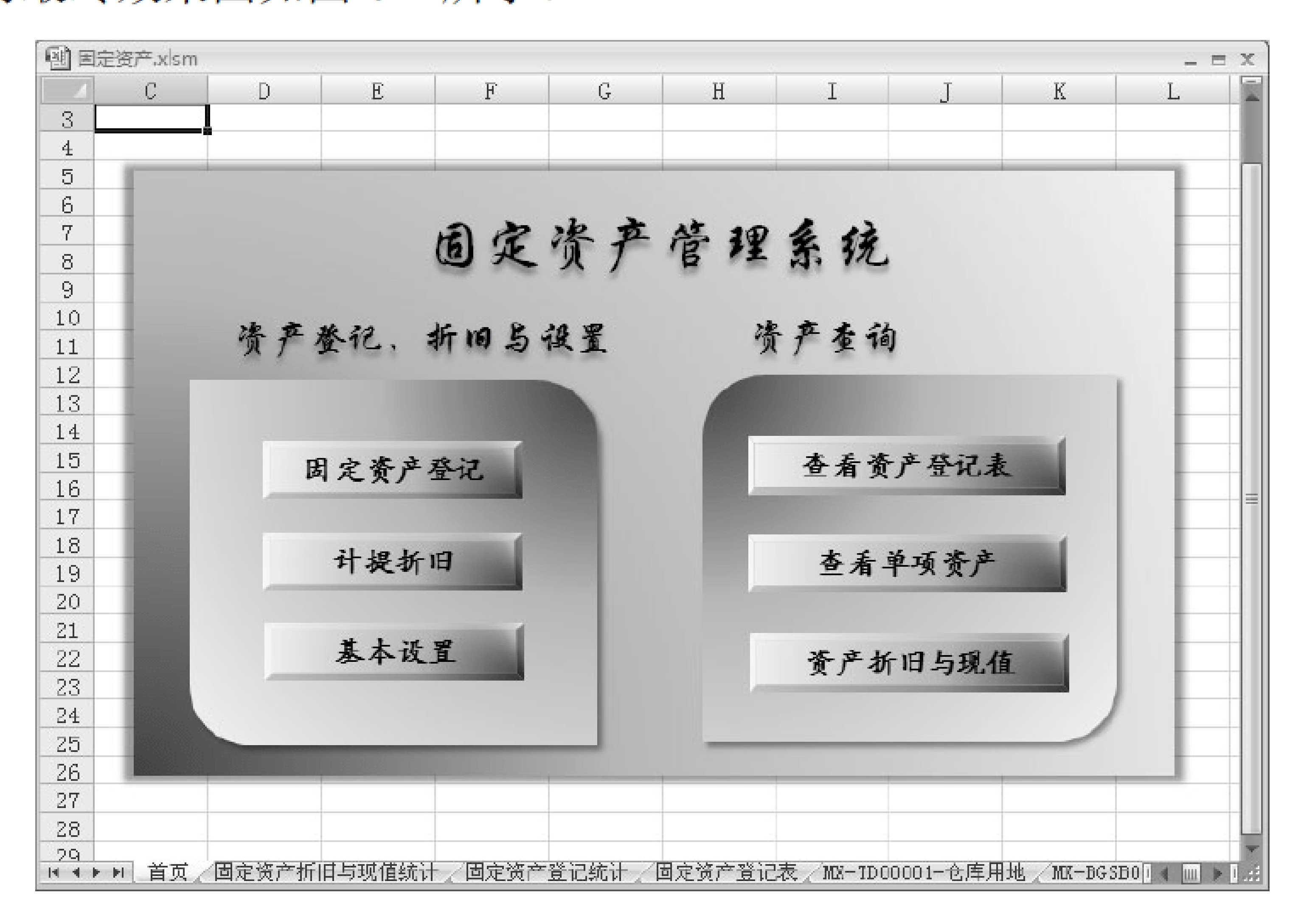


图 6-4　固定资产管理系统首页

6.2.2　首页建立步骤

建立该界面的具体步骤如下：

（1）在 Excel 2007 中依次选择【插入】|【形状】|【矩形】|【矩形】命令。然后在首页空白区域单击鼠标左键并拖动产生适当大小的矩形，如图 6-4 所示。

（2）右击刚创建的矩形，在弹出的快捷菜单中选择【设置形状格式】命令，打开【设置形状格式】对话框。在【设置形状格式】对话框中选择【填充】项并在其右侧选中【渐变填充】单选按钮，然后展开【预设颜色】下拉列表框并选择“雨后初晴”样式，如图 6-5 所示。在【类型】下拉列表框中选择“射线”项目，在【方向】下拉列表框中选择第二个样式“角度辐射”，然后展开【颜色】下拉列表框并选择“深蓝”色，如图 6-6 所示。最终填充设置如图 6-7 所示。

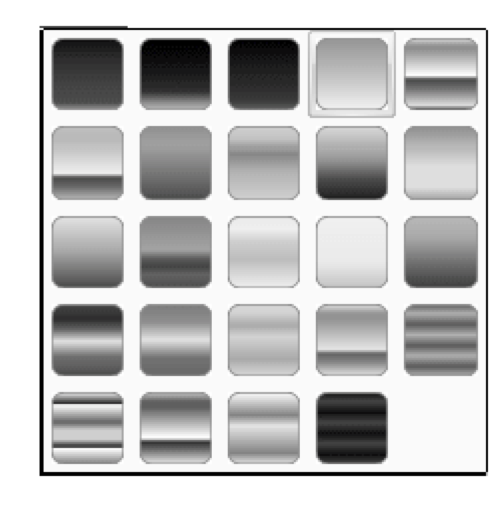


图 6-5　预设颜色样式

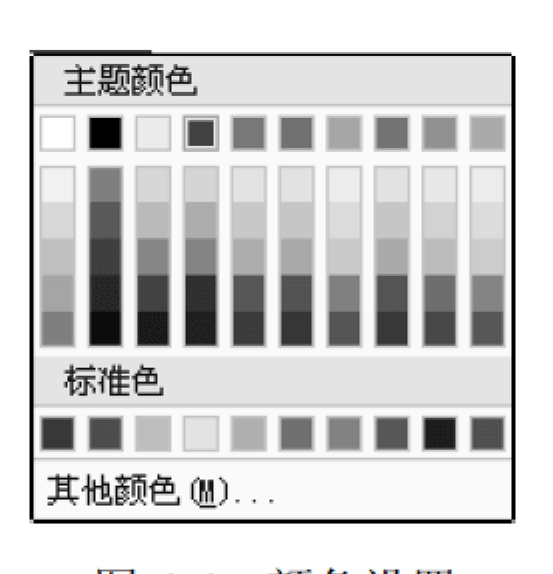


图 6-6　颜色设置

（3）在【设置形状格式】对话框中选择【线条颜色】项目。随后选中【无线条】单选按钮。

（4）在【设置形状格式】对话框中选择【阴影】项目。随后在其右侧的【预设】下拉列表框中选择【外部】分组中的【居中偏移】项目。

（5）在 Excel 2007 中依次选择【插入】|【形状】|【矩形】|【对角圆角矩形】命令。在以上创建矩形外边框的内部单击鼠标左键并拖动产生一适当大小的对角圆角矩形，使用同样的方法在矩形外边框中再创建一个对角圆角矩形。然后分别右击两对角圆角矩形，在弹出的快捷菜单中选择【编辑文字】命令，分别为其输入文字内容为“资产登记，折旧与设置”和“资产查询”。

（6）分别右击刚创建的两对角圆角矩形，在弹出的快捷菜单中选择【设置形状格式】命令，打开【设置形状格式】对话框。对角圆角矩形的格式设置与矩形外边框较为类似，不同的只是“方向”设置。“资产登记、折旧与设置”形状对应的设置为第四种角部辐射。“资产查询”形状对应的设置为第五种角部辐射。

（7）在 Excel 2007 中依次选择【插入】|【形状】|【基本】|【棱台】命令。然后在首页单击鼠标左键并拖动以产生一适当大小的棱台形状。其格式设置与前面各种形状的设置类似，这里不再赘述。

（8）复制以上创建的棱台形状 5 份。将这 6 个棱台中前 3 个依次垂直拖动到“资产登记，折旧与设置”形状。将后 3 个依次垂直拖动到“资产查询”形状。

（9）依次右击这些棱台形状，在弹出的快捷菜单中选择【编辑文字】命令。将文字内容设置为对应按钮的提示文本。随后再次右击这些形状，在弹出的快捷菜单中选择【指定宏】命令，打开【指定宏】对话框，在【指定宏】对话框的【位置】下拉列表框中选择【当前工作簿】选项，如图 6-8 所示，然后在宏名列表中选择对应的宏即可。



图 6-7　填充设置



图 6-8　指定宏

首页中各个按钮执行宏的详细列表如表 6-1 所示。

表 6-1　首页按钮执行宏列表

按钮标签	执行宏
固定资产登记	固定资产登记
计提折旧	计提折旧
基本设置	基本设置
查看资产登记表	固定资产统计
查看单项资产	折旧明细表
资产折旧与现值	折旧与现值统计

6.3　其他无代码表设计

除首页外，还有两个工作表也是无代码工作表，分别是单项固定资产折旧明细模板工作表、设置表工作表。这两个工作表的界面都比较简单。设计工作没有触及形状或是控件，仅仅是填写数据和调整单元格格式。因此将这两个表的设计合并到一节讲述。

6.3.1　单项固定资产折旧明细模板表设计

单项固定资产折旧明细模板工作表大致可以划分为两个部分，一个是固定资产登记信息区，一个是固定资产折旧明细账区，另外还有一个返回按钮。该表的界面如图 6-9 所示。

在固定资产登记信息区中，显示的是在固定资产登记时获取的该固定资产的所有基本信息，其中年折旧率、年折旧额、残值、月折旧率、月折旧额是公式自动产生的。

在固定资产折旧明细账区中，显示的是该固定资产进行折旧操作时可能涉及到的各个项目。其中性质栏是用于提示是否固定资产折旧计提完毕。当固定资产折旧计提完毕后，该栏显示“计提完毕”，否则不显示任何信息。在工作表中只显示了 N 列数据，其他的列都被隐藏了。

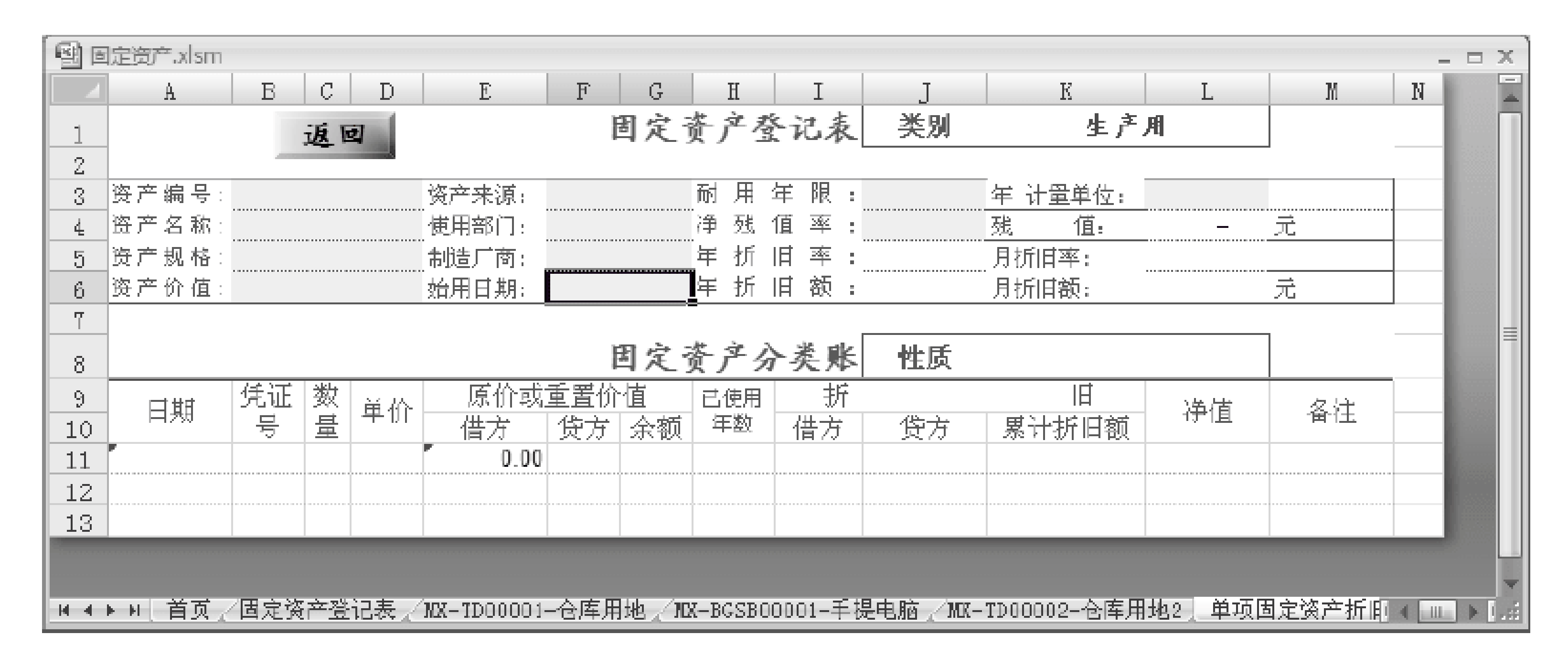


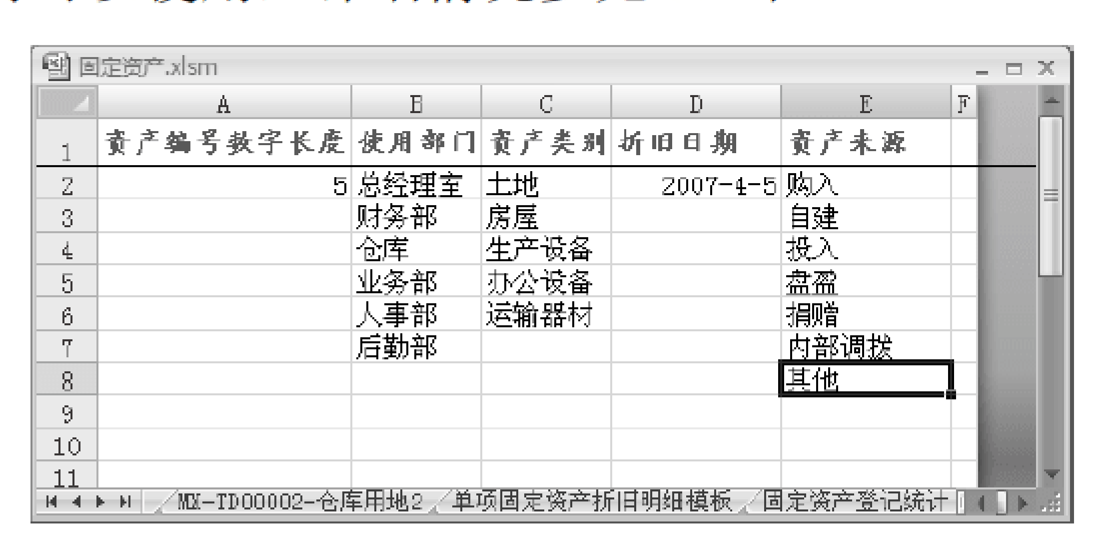
图 6-9　固定资产折旧明细模板界面

6.3.2　设置表工作表设计

设置表用于存储系统在运行中需要用到的一些基础设置信息。该表的界面如图 6-10 所示。

这些信息包括资产编号数字长度、使用部门、资产类别、折旧日期、资产来源。以下是这些基础设置信息的详细解释：

- 资产编号数字长度：本系统中对于固定资产的资产编号采用“资产类别头字拼音+数字标识”的方式。该设置就是设置数字标识段的总长度。
- 使用部门：存储所有可能使用固定资产的部门资料。该资料在固定资产登记表中输入时可以使用，详细情况参见 6.4 节。
- 资产类别：和使用部门的意义相似，它存储可能的固定资产分类。在固定资产登记表中输入时可以使用，详细情况参见 6.4 节。
- 折旧日期：该日期为最后一次进行计提折旧时在“frm 计提日期”窗体中设置的折旧日期。
- 资产来源：和使用部门的意义相似，它存储可能的固定资产来源方式。在固定资产登记表中输入时可以使用，详细情况参见 6.4 节。



资产编号数字长度	使用部门	资产类别	折旧日期	资产来源
5	总经理室	土地	2007-4-5	购入
	财务部	房屋		自建
	仓库	生产设备		投入
	业务部	办公设备		盘盈
	人事部	运输器材		捐赠
	后勤部			内部调拨
				其他

图 6-10　设置表

6.4　固定资产登记表设计

在使用该系统进行固定资产管理时，首先要在系统中建立各个固定资产的信息，后续的所有操作都是基于这些已建立的数据而进行的。该步骤在整个管理过程中处于源头，因此在此需要保证数据建立的正确性。

6.4.1　表界面设计

该表的界面如图 6-11 所示，其中包含了 3 个棱台形状按钮，分别执行返回、利用数据和保存功能。这 3 个按钮的具体功能描述如下：

- 【返回】按钮：返回到首页。
- 【利用数据】按钮：利用该按钮可以使用已经建立的固定资产数据填充该登记表。如果需建立的固定资产的信息与已建立的固定资产较为相似，可以通过该按钮快速建立。需要注意的是该按钮直接填充数据中还包含了资产编号，因为资产编号不能重复，在保存前需要修改该资产编号。
- 【保存】按钮：该按钮完成两个工作，一个是将新登记的固定资产信息保存到以“MX-”

+资产编号+资产名称格式命名的工作表中，另外它还将会将这些信息也写入固定资产登记表中。



图 6-11 固定资产登记表界面

在工作表中有一部分单元格的内容是自动生成的。这些单元格是年折旧率（J5）、年折旧额（J6）、残值（L4）、月折旧率（L5）和月折旧额（L6）5 个单元格。这些公式的基础数据来源于耐用年限（J3）和净残值率（J4）。这些单元格的公式如表 6-2 所示。

表 6-2 单元格公式列表

单 元 格	公 式
年折旧率（J5）	=IF(J3<>"",(1-J4)/J3,"")
年折旧额（J6）	=IF(J3<>"",B6*J5,"")
残值（L4）	=B6*J4
月折旧率（L5）	=IF(J5<>"",J5/12,"")
月折旧额（L6）	=IF(L5<>"",ROUND(B6*L5,2),"")

在该表中建立固定资产信息时，可以通过利用数据按钮借用已建立的固定资产信息。还有部分单元格的数据可以通过双击该单元格获得，这些单元格包括类别（K1）、资产来源（F3）、使用部门（F4）。双击对应单元格后程序会弹出一个选择信息框。详细介绍见本节的代码设计。

6.4.2 设置单元格条件格式

在图 6-11 中，单元格显示有填充颜色的均设置了条件格式。该条件格式的目的在于当这些单元格中没有输入任何数据时，对填充颜色进行修改，以起到提示作用。设置的方式如下：

首先在固定资产登记表中将这些单元格都一起选中。然后依次选择【开始】|【条件格式】|【新建规则】命令，弹出【新建格式规则】对话框，在【选择规则类型】列表框中选择【只为包含以下内容的单元格设置格式】项目。条件设置为【单元格值】|【等于】|【0】。单击【格式】按钮设置填充颜色格式后确认。最后的新建格式规则设置如图 6-12 所示。



图 6-12　新建格式规则设置

6.4.3　表初始化代码

在固定资产登记表被激活时，需要将该表中需要填写信息的单元格内容清除，以便输入新的数据。本实例在该表的初始化事件过程中通过一个“重置登记表”过程完成该项工作。

在该过程中，程序首先通过一个数组保存了工作表中部分单元格的地址，然后通过一个循环，遍历数组各个元素，将需要进行修改的单元格内容清空。在数组保存的单元格中，只有一部分单元格是通过公式自动产生数据的，而其余的单元格都不需要通过代码实现数据修改，比如 J5、J6、L4 和 L5。该过程的流程图如图 6-13 所示。

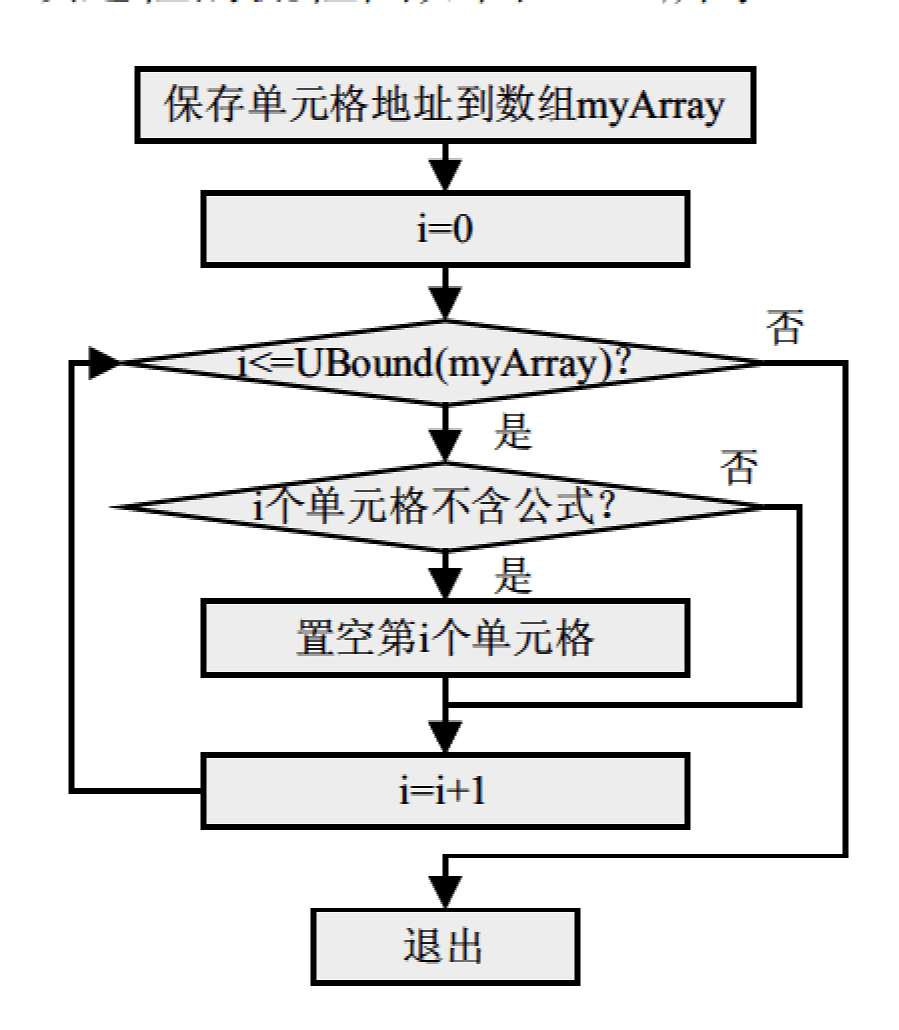


图 6-13　重置登记表过程流程图

以下是该工作表激活事件代码及其调用的过程的详细代码解释：

```
Private Sub Worksheet_Activate()
重置登记表                                         '清空需要填写资料的单元格
End Sub

Sub  重置登记表()
```

```
Dim myArray() As Variant, i As Integer
'定义单元格数组，方便循环清空单元格内容
myArray = Array("K1", "B3", "B4", "B5", "B6", "F3", "F4", "F5", "F6", _
                "J3", "J4", "J5", "J6", "L3", "L4", "L5", "L6")
For i = 0 To UBound(myArray)
    '循环检测各个单元格，当不是自动产生数据的单元格时，将该单元格内容设置为空
    If myArray(i) <> "J5" And myArray(i) <> "J6" And myArray(i) <> "L4" And myArray(i) <> _
        "L5" And myArray(i) <> "L6" Then
        Range(myArray(i)) = Empty                    '置空单元格内容
    End If
Next
End Sub
```

6.4.4　工作表双击事件代码

该事件过程用于打开输入辅助提示框。当用户在表中双击时，检测发生双击的对象是否需要弹出对应单元格的输入提示框。表中单元格，包括类别（K1）、资产来源（F3）、使用部门（F4）被双击时，都会弹出该窗口，而弹出的窗口显示的可选内容也会根据不同的情况发生变化。

当确认了窗体需要显示的项目类型后，程序将该类型设置情况保存到公共变量“辅助窗口参数”中。打开输入辅助窗体后，窗体将根据该参数确定显示项目类型。用户在窗体中选择了项目后，该选择项目的信息仍然被存储在“辅助窗口参数”公共变量中。然后程序将该信息填写到双击单元格。为了避免双击造成的单元格进入编辑状态。程序通过 SendKey 方法发送了一个 Esc 按键动作来退出单元格编辑状态。

如果用户双击类别单元格并且选择了资产类别项目后，程序将调用获取资产编号函数，该函数根据用户的选择项目确定当前新建立资产的编号。

有关输入辅助窗体的设计过程见窗体设计节的输入辅助窗体小节。另外在该工作表的双击事件过程中，调用了“获取资产编号”函数，该函数的介绍请见公共模块节。以下是双击事件的详细代码解释：

```
Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)
'检测是否双击了资产类别输入单元格
If Target.Row = 1 And (Target.Column = 11 Or Target.Column = 12) Then
    辅助窗口参数 = "资产类别"              '该公共变量在输入辅助窗体中将会调用
    frm 输入辅助.Show                      '显示输入辅助窗口，窗口被关闭时会修改辅助窗口参数
    资产登记表.Range("K1").Value = 辅助窗口参数      '获得输入的资产类别值
    Application.SendKeys "{ESC}"                   '退出单元格的编辑状态
    '当获取了资产类别时，自动产生资产编号，并写入资产编号编辑单元格中
    If Len(资产登记表.Range("K1")) Then
        资产登记表.Range("B3") = 获取资产编号(资产登记表.Range("K1"))
    End If
End If
'检测是否双击了资产来源输入单元格
If Target.Row = 3 And (Target.Column = 6 Or Target.Column = 7) Then
```

```
        辅助窗口参数 = "资产来源"
        frm 输入辅助.Show
        资产登记表.Range("F4").Value = 辅助窗口参数
        Application.SendKeys "{ESC}"
End If
'检测是否双击了使用部门输入单元格
If Target.Row = 4 And (Target.Column = 6 Or Target.Column = 7) Then
        辅助窗口参数 = "使用部门"
        frm 输入辅助.Show
        资产登记表.Range("F3").Value = 辅助窗口参数
        Application.SendKeys "{ESC}"
End If
End Sub
```

6.4.5 固定资产保存

当在固定资产登记表中完成了某个固定资产的信息填入工作后，需要将该信息填入相应的固定资产折旧明细表和固定资产登记统计表中。这个工作由“固定资产登记保存”过程完成。当单击按钮保存时，该过程被执行。

在固定资产登记表中有些项目是必填项目，包括资产类别、资产编号、资产名称、资产价值、使用部门、始用时间、使用年限。该过程首先检查这些项目是否为空，然后检查资产编号是否有重复，最后保存并显示保存进度。该过程的详细流程图如图 6-14 所示。

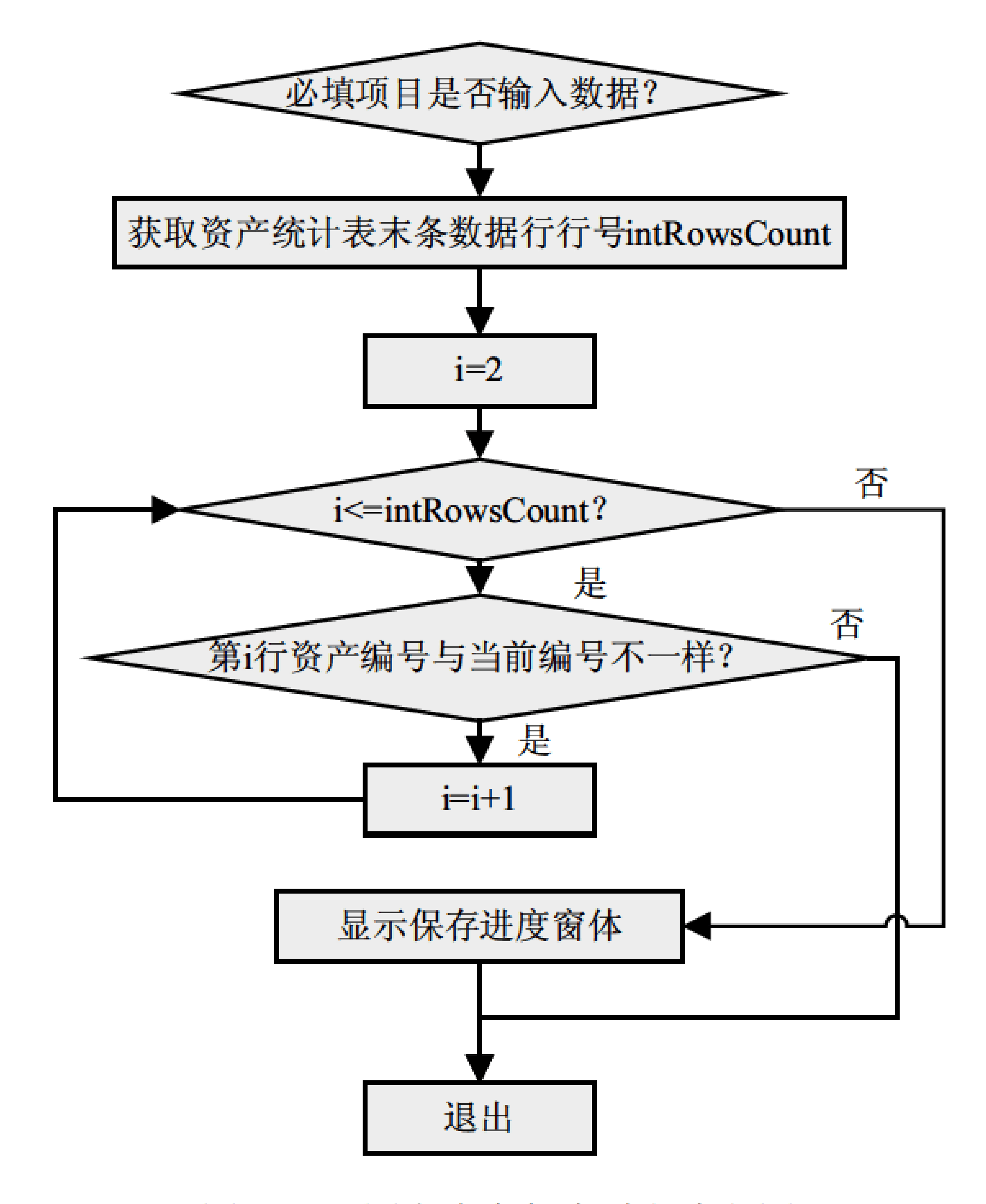


图 6-14　固定资产保存过程流程图

以下是该过程的详细代码解释：

```
Sub 固定资产登记保存()
Dim intRowsCount As Integer, i As Integer
'依次检查各个必填项目，当未填写内容时，提示未填写项目信息，然后终结过程执行
If Len(Range("K1")) = 0 Then
    MsgBox "资产类别不能为空！", vbOKOnly + vbInformation          '提示资产类别不能为空
    Exit Sub
ElseIf Len(Range("B3")) = 0 Then
    MsgBox "资产编号不能为空！", vbOKOnly + vbInformation          '提示资产编号不能为空
    Exit Sub
ElseIf Len(Range("B4")) = 0 Then
    MsgBox "资产名称不能为空！", vbOKOnly + vbInformation          '提示资产名称不能为空
    Exit Sub
ElseIf Len(Range("B6")) = 0 Then
    MsgBox "资产价值不能为空！", vbOKOnly + vbInformation          '提示资产价值不能为空
    Exit Sub
ElseIf Len(Range("F4")) = 0 Then
    MsgBox "使用部门不能为空！", vbOKOnly + vbInformation          '提示使用部门不能为空
    Exit Sub
ElseIf Len(Range("F6")) = 0 Then
    MsgBox "始用时间不能为空！", vbOKOnly + vbInformation          '提示始用时间不能为空
    Exit Sub
ElseIf Len(Range("j3")) = 0 Then
    MsgBox "使用年限不能为空！", vbOKOnly + vbInformation          '提示使用年限不能为空
    Exit Sub
End If
'检查资产编号是否存在重复项目
'将当前的资产编号与资产登记统计表中的资产编号列中所有编号比对，当找到有一致时，提示编号重复，
 然后退出
intRowsCount = 资产登记统计.Range("B" & Rows.Count).End(xlUp).Row
For i = 2 To intRowsCount
    If 资产登记统计.Range("C" & i) = Range("B3") Then
        MsgBox "资产编号和已有资产冲突！", vbOKOnly + vbInformation
        Exit Sub
    End If
Next
'开始保存工作，并显示保存进度窗体
frm 进度.Show
End Sub
```

6.5　固定资产登记统计表设计

固定资产登记统计工作表用于存储所有已完成登记且保存过的固定资产登记信息。用户也可以通过该工作表打开对应固定资产的详细折旧明细表，查看单条的固定资产信息。为了便于用户在该工作表中浏览固定资产的详细信息，工作表的前 3 列都被冻结。

6.5.1 界面设计

固定资产登记统计工作表的界面设计比较简单。在表中 A1 单元格有一个返回按钮，用户通过该按钮可以跳转回首页界面。固定资产各个详细信息以列表的形式体现出来。在该表中浏览时，为了保证有些重要信息能够固定下来，需要冻结窗格。

这些需要冻结的内容包括【返回】按钮列、资产类别列、资产编号列和资产名称列。操作方法是：首先选择 E2 单元格，然后依次选择【视图】|【冻结窗口】|【冻结拆分窗格】命令。如图 6-15 所示显示了已经登记了两个固定资产后的固定资产登记统计表。

A	B	C	D	E	F	G	H	I	J	K
返回	资产类别	资产编号	资产名称	资产规格	资产价值	资产来源	使用部门	制造厂商	始用日期	耐用年限
	土地	TD00001	仓库用地	144m²	300000	自建	仓库	我司	2006-4-10	30
	办公设备	BGSB00001	手提电脑	IBM	10000	购入	总经理室		2007-1-10	5
	运输器材	YSQC00001	轿车	桑塔纳3000	100000	购入	总经理室	我司	2007-4-10	20

图 6-15 固定资产登记统计

在该工作表中包含的固定资产信息列比较多。用户可能在该模式下查看单个固定资产信息时感觉不习惯，此时用户可以双击固定资产信息行中某个单元格，随后程序会自动跳转到该固定资产的详细情况表中，如图 6-16 所示。该图显示的是在固定资产登记统计工作表中双击最后一条记录时的显示结果。此时再单击其中的【返回】按钮，将会跳转到固定资产登记统计工作表中。



返回　固定资产登记表　类别　运输器材

资产编号：	YSQC00001	资产来源：	购入	耐用年限：	20 年
资产名称：	轿车	使用部门：	总经理室	净残值率：	5%
资产规格：	桑塔纳3000	制造厂商：	我司	年折旧率：	4.7500%
资产价值：	100,000.00	始用日期：	2007-4-10	年折旧额：	4,750.00

计量单位：	
残值：	5,000.00 元
月折旧率：	0.3958%
月折旧额：	395.83 元

固定资产分类账　性质

日期	凭证号	数量	单价	原价或重置价值 借方	贷方	余额	已使用年数	折旧 借方	贷方	累计折旧额	净值	备注
2007-4				100,000.00								
2007-12									395.83	395.83	99,604.17	

图 6-16 查看详细固定资产信息

6.5.2 代码设计

在该工作表中双击标题行以下的某个单元格时，程序会检测当前被双击单元格所在行是否有固定资产记录。如果存在，将会打开该固定资产的折旧明细表，同时程序还设置了单击【返回】按钮时的返回位置，如果没有记录数据，则不执行任何操作。

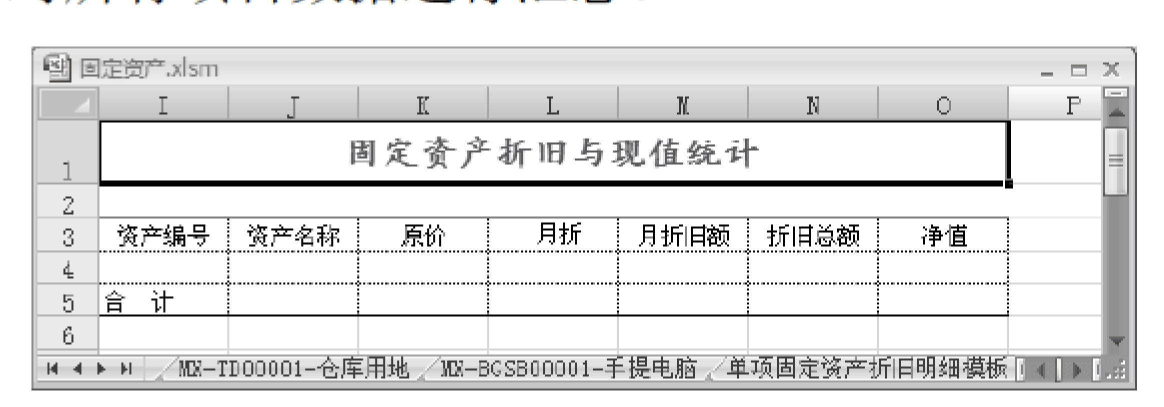
在该过程中用到了“是否返回统计表”变量。当该变量被设置为真时，单击【返回】按钮将返回到固定资产登记统计表中，否则返回到首页。

以下是该工作表的事件过程的详细代码解释：

```
Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)
Dim intRowsCount As Integer
'获取资产登记统计工作表末条资产所在行号
intRowsCount = 资产登记统计.Range("B" & Rows.Count).End(xlUp).Row
If Target.Row >= 2 And Target.Row <= intRowsCount Then          '检测被双击单元格所在行是否有
                                                                  记录数据
    是否返回统计表 = True                                        '设置单击返回按钮时返回位置
    '激活所选固定资产的详细信息工作表
    Sheets("MX-" & Range("C" & Target.Column) & "-" & Range("D" & Target.Column)).Select
End If
End Sub
```

6.6　固定资产折旧与现值表设计

固定资产折旧与现值表是通过一个表格的形式完成对各项固定资产折旧额和现值、固定资产总折旧和总现值的统计工作。该表的格式被隐藏在 I 列与 O 列之间。其格式包含标题、表头、一行空行以及合计行（如图 6-17 所示）。当该表被激活时，系统将该格式复制到 A 列到 G 列。然后循环各个固定资产折旧明细表，获得该固定资产的各项数据后插入到空行数据中。最后的一项合计行对所有项目数据进行汇总。



	I	J	K	L	M	N	O
1	固定资产折旧与现值统计						
2							
3	资产编号	资产名称	原价	月折	月折旧额	折旧总额	净值
4							
5	合　计						

图 6-17　固定资产折旧与现值统计格式

6.6.1　表界面设计

该表被激活后的显示界面比较简洁：通过按钮可以返回系统首页。该表的数据项目是通过工作表的激活事件实现重新计算的，即每次该表被激活时，该表都会重新产生一次。当数据固定资产折旧明细表发生变化时，其统计数据可以通过再次激活该表得到刷新。该表最终设计完成后的界面如图 6-18 所示。

	A	B	C	D	E	F	G
1	固定资产折旧与现值统计						
2							
3	资产编号	资产名称	原价	月折	月折旧额	折旧总额	净值
4	TD00001	仓库用地	400,000.00	0.26%	1,055.56	8444.48	391555.52
5	BGSB00001	手提电脑	10,000.00	1.62%	161.67	485.01	9514.99
6	合　计		410,000.00		1,217.23	8,929.49	401,070.51

图 6-18　固定资产折旧与现值统计表

另外该表从 H 列以后的数据都被隐藏掉。在该表的 I1 到 O5 单元格区域中保存了表的头

部格式。每次工作表被激活时，清除掉表中显示数据后，程序都会从该隐藏区域中获取头部数据与格式。

6.6.2 表代码设计

该表的代码包含在两个事件过程代码中。一个是表被激活时的 Worksheet_Activate 事件，一个是表脱离激活状态的 Worksheet_Deactivate 事件。这两个事件的功能描述如下：

- 表激活事件：表被激活时的 Worksheet_Activate 事件用于重新获取数据，形成固定资产折旧与现值统计数据。
- 表失去激活事件：表脱离激活状态的 Worksheet_Deactivate 事件用于清除该表中所有数据。这些数据不包括单元格的格式设置。

这两个事件中表失去激活事件过程比较简单，代码也很精简，这里不再针对该事件多做说明。以下重点讲述表激活事件过程的流程及其流程图。

当工作表被激活时，程序首先获取了资产登记统计表末条数据的行号。然后根据该值确定是否有固定资产记录被登记。当确认有固定资产登记时，程序将依次插入对应行数，以统计各个固定资产的折旧与现值。然后把这些固定资产的信息依次写入新添加的行中。最后程序通过赋予单元格计算公式完成各项统计工作。这些工作表包括原价汇总、月折旧额汇总、折旧总额汇总和净值汇总。如图 6-19 所示的是该事件过程的流程。

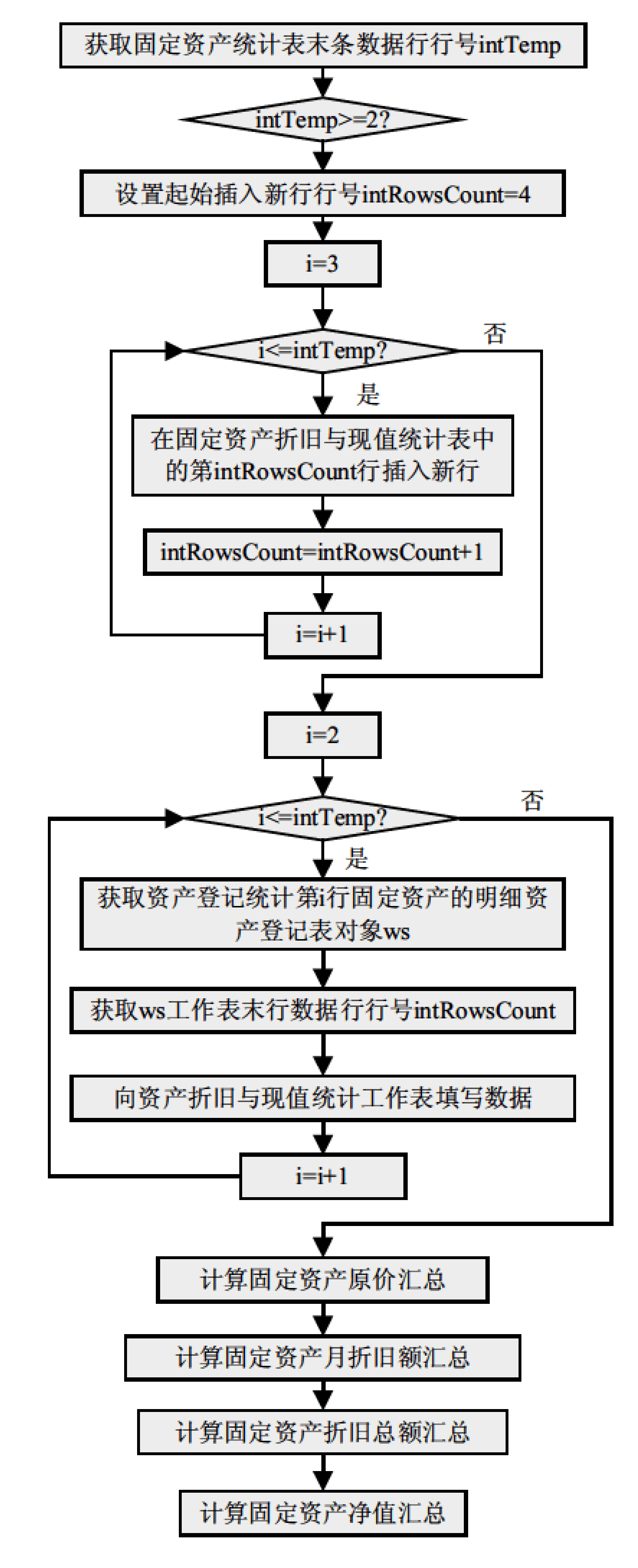


图 6-19　固定资产折旧与现值统计表激活事件过程流程图

这两个事件的详细代码解释如下：

```
Private Sub Worksheet_Activate()
Dim intRowsCount As Integer, i As Integer, intTemp As Integer
Dim ws As Worksheet
'复制 I1 到 O5 区域的单元格，粘贴到 A1 单元格，该区域保存的是表格的格式
Range(Cells(1, 9), Cells(5, 15)).Copy Range("A1")
'获取固定资产统计表的有数据行的末行号
intTemp = 资产登记统计.Range("B" & Rows.Count).End(xlUp).Row
```

```
'如果在固定资产统计表中有固定资产记录存在，即有数据行的行号至少大于 2 时，开始填充表中空行数据区
If intTemp >= 2 Then
    intRowsCount = 4
    '以下循环用于产生足够多的空行，因为首行空行已经存在且 intTemp 记录的是包括固定资产统计表标题行的行号,所以循环变量起始值为 3，终值为 intTemp
    For i = 3 To intTemp
        '在最后总计行单元格区域上插入一行空行数据，且总计行向下移动一行
        Range(Cells(intRowsCount, 1), Cells(intRowsCount, 7)).Insert shift:=xlShiftDown
        '保存当前总几行的行号
        intRowsCount = intRowsCount + 1
    Next
    '循环各个固定资产折旧明细表，获取相应数据
    For i = 2 To intTemp
        '通过固定资产登记统计表中的记录，获取表名，根据表名建立各个固定资产折旧明细表对象
        Set ws = Worksheets("MX-" & 资产登记统计.Range("C" & i) & "-" & 资产登记统计.Range("D" & i))
        '获取当前固定资产折旧明细表的最大行号
        intRowsCount = ws.Range("A" & Rows.Count).End(xlUp).Row
        '根据固定资产折旧明细表中的数据填充固定资产折旧与现值表相应的栏位
        With ws
            Range("A" & (i + 2)) = .Range("B3")                    '资产编号
            Range("B" & (i + 2)) = .Range("B4")                    '资产名称
            Range("C" & (i + 2)) = .Range("B6")                    '原价
            Range("D" & (i + 2)) = .Range("L5")                    '月折
            Range("E" & (i + 2)) = .Range("L6")                    '月折旧额
            Range("F" & (i + 2)) = .Range("K" & intRowsCount)      '折旧总额
            Range("G" & (i + 2)) = .Range("L" & intRowsCount)      '净值
        End With
    Next
    '形成汇总行数据
    intRowsCount = Range("A" & Rows.Count).End(xlUp).Row      '获取汇总行行号
    '原价汇总
    Range("C" & intRowsCount).Formula = "=sum(C4:C" & (intRowsCount -1) & ")"
    '月折旧额汇总
    Range("E" & intRowsCount).Formula = "=sum(E4:E" & (intRowsCount -1) & ")"
    '折旧总额汇总
    Range("F" & intRowsCount).Formula = "=sum(F4:F" & (intRowsCount -1) & ")"
    '净值汇总
    Range("G" & intRowsCount).Formula = "=sum(G4:G" & (intRowsCount -1) & ")"
    Set ws = Nothing
End If
End Sub

Private Sub Worksheet_Deactivate()
Dim intRowsCount As Integer
intRowsCount = Range("A" & Rows.Count).End(xlUp).Row
'清除 A 列到 G 列的有数据区域单元格
Range(Cells(1, 1), Cells(intRowsCount, 7)).Delete shift:=xlShiftUp
End Sub
```

6.7 基本设置窗体设计

基本设置窗体为系统提供了设置系统基本信息的界面。对于该系统，在运行之前，需要建立其相应的基本信息资料，信息包括资产编号、使用部门、资产类别、资产来源。这些资料都被最终保存在设置表中，以便于系统其他功能模块调用。

由于该窗体的代码比较多，在介绍窗体代码设计时，笔者将这些代码进行了分类。详细情况参见本节的后续小节内容。

6.7.1 窗体界面设计

由于在该窗体中需要设计的内容繁多，因此采用了多页控件，其中资产编号、使用部门、资产类别、资产来源各占用一个页。表 6-3 列出了该窗体下除标题控件外所有控件的控件名、所属页控件名、功能信息。

表 6-3 基本设置表控件列表

控　件　名	所属页控件名	控 件 说 明
数字长度	资产编号	文本框控件。完成显示或输入操作，其操作的对象是设置表中资产编号长度即 A2 单元格
部门列表	使用部门	列表框控件。显示当前在设置表中存在的所有部门
部门名称	使用部门	文本框控件。显示当前在部门列表中选择的部门或输入新部门
部门添加	使用部门	按钮。单击该按钮将当前部门名称文本框中的部门添加到设置表的使用部门中
部门修改	使用部门	按钮。将当前在部门列表中选择的部门修改为部门名称中的新部门名
部门删除	使用部门	按钮。将在部门名称中显示的部门删除掉
类别列表	资产类别	列表框控件。显示当前在设置表中存在的所有类别
类别名称	资产类别	文本框控件。显示当前在类别列表中选择的类别或输入新类别
类别添加	资产类别	按钮。单击该按钮将当前类别名称文本框中的类别添加到设置表的使用类别中
类别修改	资产类别	按钮。将当前在类别列表中选择的类别修改为类别名称中的新类别名
类别删除	资产类别	按钮。将在类别名称中显示的类别删除掉
来源列表	资产来源	列表框控件。显示当前在设置表中存在的所有来源
来源名称	资产来源	文本框控件。显示当前在来源列表中选择的来源或输入新来源
来源添加	资产来源	按钮。单击该按钮将当前来源名称文本框中的来源添加到设置表的资产来源中
来源修改	资产来源	按钮。将当前在来源列表中选择的来源修改为来源名称中的新来源名
来源删除	资产来源	按钮。将在来源名称中显示的来源删除掉
确定	frm 基本设置	按钮。单击该按钮将当前设置保存到设置表中
关闭	frm 基本设置	按钮。单击该按钮退出该窗体，并且不保存设置

在页控件中，资产编号页界面效果如图 6-20 所示。使用部门、资产类别、资产来源 3 页

的布局设计大体一致，因此这 3 页的效果图仅给出使用部门设置图。使用部门的界面效果图如图 6-21 所示。

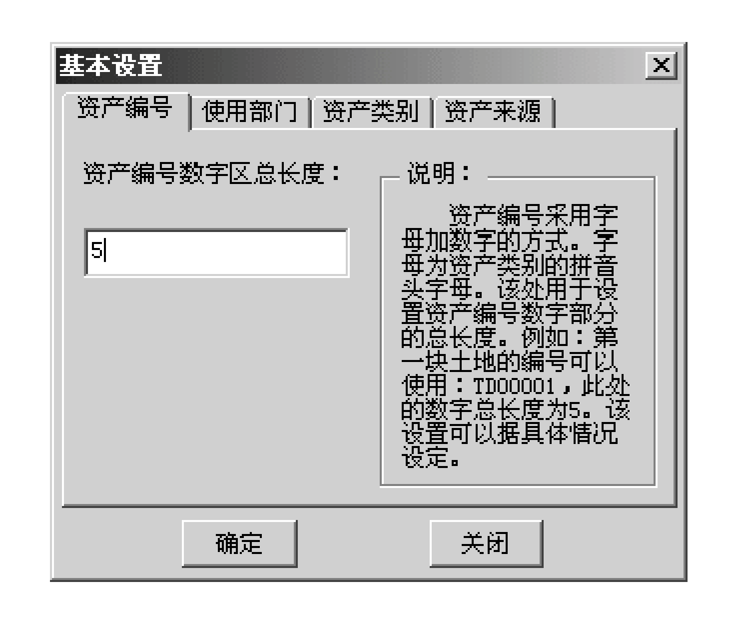


图 6-20　资产编号设置



图 6-21　使用部门设置

要建立该窗口，可参照步骤操作。由于使用部门页和资产类别、资产来源页结构类似。以下介绍步骤时，只对资产编号与使用部门页的建立做详细介绍。其他两个页面用户请参照步骤中建立使用部门页面的步骤完成。

（1）在 Excel 2007 VBE 开发环境中依次选择【插入】|【用户窗体】命令。在属性窗口中设置新插入窗体的名称属性为“frm 基本设置”，其他属性保持默认即可，如图 6-22 所示。

（2）在工具箱中选择多页控件，然后在窗体中插入一多页控件，此时新建立的多页控件默认包含了两个页。在属性窗口依次将这两个页的名称属性修改为“资产编号”和“使用部门”。并将其 Caption 属性设置成与其名称属性一致，如图 6-23 所示。



图 6-22　基本设置窗体属性设计



图 6-23　“资产编号”页设计效果

（3）单击多页控件中的“资产编号”页，然后在工具箱中选择框架控件。在该页的右侧插入一个框架控件，随后在属性窗口中修改该框架的 Caption 属性为“说明：”，如图 6-23 所示。

（4）在工具箱中选择标签控件。在“资产编号”标签的左上侧与框架控件内部各插入一标签控件，然后在属性窗口中修改这两个标签的 Caption 属性，第一个标签的 Caption 属性为“资产编号数字区总长度：”，但另外一个标签的 Caption 字符串很长，这里不再给出该字符串数据，如图 6-23 所示。

（5）在工具箱中选择文本框控件。在“资产编号数字区总长度：”标签的下方插入一文本框控件。然后在属性窗口中修改该文本框的名称属性为“数字长度”。

（6）在多页控件中单击“使用部门”页，然后在工具箱中选择框架控件。在“使用部门”页的左侧插入一个框架控件。然后在属性窗口中修改该框架的 Caption 属性为“已有部门：”，如图 6-24 所示。

（7）在工具箱中选择列表框控件。在上一步创建的框架控件内插入一列表框控件，然后在属性窗口中修改该控件的名称属性为“部门列表”。

（8）在工具箱中选择标签控件。在“使用部门”页的右上方插入一标签控件。然后在属性窗口修改该标签的 Caption 属性为“部门名称：”。

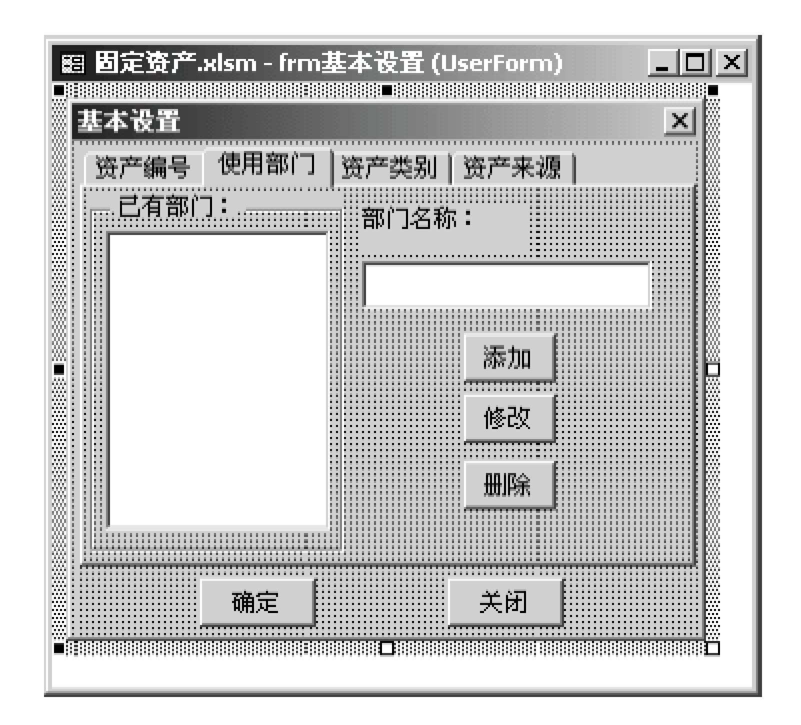


图 6-24　“使用部门”页设计效果

（9）在工具箱中选择文本框控件。在上一步创建的标签控件下面插入一文本框控件。然后在属性窗口修改该文本框的名称属性为“部门名称”。

（10）在工具箱中选择按钮控件。在刚插入的文本框下方连续插入 3 个按钮控件。然后在属性窗口中依次修改这些按钮的 Caption 属性为“添加”、“修改”和“删除”，并将其名称属性依次修改为“部门添加”、“部门修改”和“部门删除”。

（11）复制“使用部门”页两次。然后在属性窗口中分别将其名称属性修改为“资产类别”和“资产来源”，并将其 Caption 属性设置为与名称属性一样，如图 6-25 和图 6-26 所示。

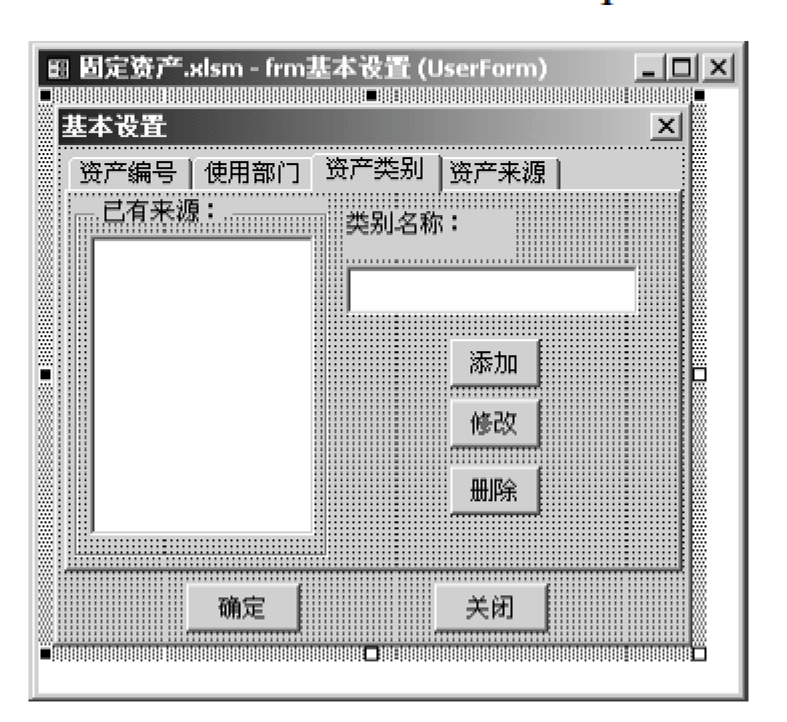


图 6-25　“资产类别”页设计效果

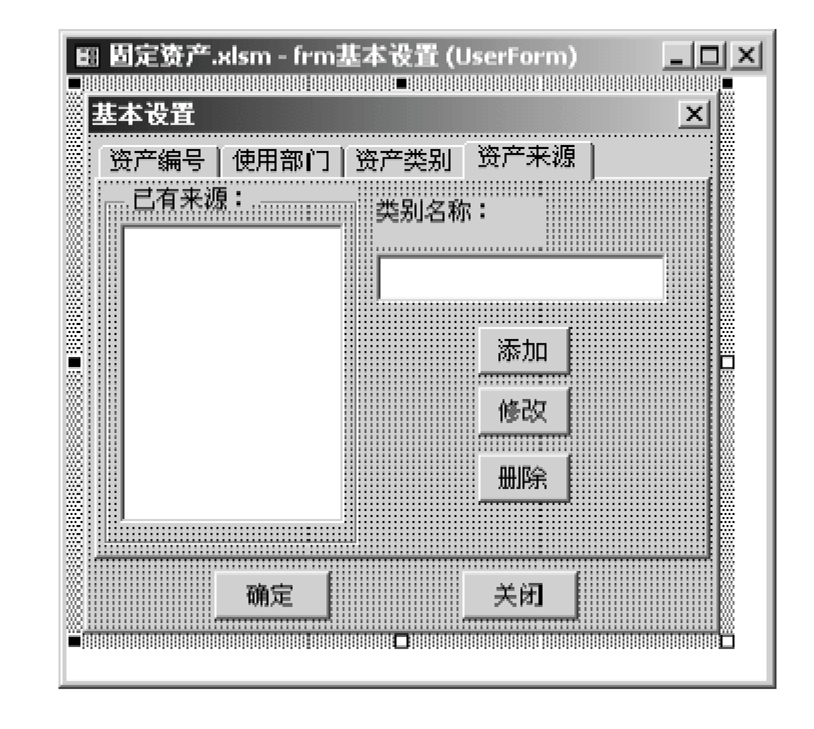


图 6-26　“资产来源”页设计效果

（12）在工具箱中选择按钮控件。在窗体的底部连续插入两个按钮控件，然后在属性窗口中修改按钮的 Caption 属性分别为“确定”和“关闭”，并修改按钮的名称属性依次为“确定”和“关闭”。

6.7.2　窗体初始化与确定、关闭按钮代码设计

本小节介绍窗体初始化事件以及确定和关闭按钮的代码设计。在窗口初始化时，需要完成的工作包括两件事情，分别是设置资产编号页为初始显示页、完成对资产编号页中控件的初始显示。【确定】按钮被单击时，程序需要检测新设置的数字长度设置是否与原来的设置一致，并且根据检测的结果决定工作簿是否需要保存。【关闭】按钮完成的事情十分简单，只需要退出窗口即可。

另外在该窗体中还定义了一个局部变量 IsConfigChange。该变量用于记录当前的设置是否发生变化，以便于确认该设置表是否需要保存。本小节的代码十分简单，这里不再给出过程的流程图。以下是窗体初始化代码、确定和关闭按钮的单击事件代码的详细代码解释：

```
Private Sub UserForm_Initialize()
'将资产编号页指定为窗体初始显示时，多页控件初始页
MultiPage1.Value = 0
'调用资产编号页初始化过程
初始化资产编号页
End Sub

Private Sub 确定_Click()
'检查资产编号页中数字长度文本框的最终值是否与设置表的值一致，不一致时保存结果
If 数字长度.Text <> 设置表.Cells(2, 1) Then                '检测新数字长度设置是否与原设置一致
    设置表.Cells(2, 1) = 数字长度.Text                      '修改原数字长度设置
    IsConfigChange = True                                   '标记工作簿需要保存
End If
'检查 IsConfigChange 变量，如果是真时，保存工作簿
If IsConfigChange Then                                      '检测工作簿是否需要保存
    ThisWorkbook.Save                                       '保存工作簿
End If
Unload Me
End Sub

Private Sub 关闭_Click()
Unload Me
End Sub
```

6.7.3　初始化页过程代码解释

在窗口的页控件中包含了 4 个页。当这 4 页被显示时，都首先需要对该页中所包含控件进行部分设置。程序通过 4 个过程分别完成这 4 页的初始显示设置，这 4 个过程分别是初始化资产编号页、初始化使用部门页、初始化资产类别页和初始化资产来源页。其中后面 3 个过程完成工作十分相似。以下对这 4 个过程分为两类介绍其功能：

- 初始化资产编号页：资产编号页被首次显示时需要从工作簿的设置工作表中读取用户设置的数字长度信息。该项设置用于确定固定资产编号中数字的位数。
- 初始化其他页：初始化使用部门页、初始化资产类别页和初始化资产来源页完成工作大体相似。这些被首次显示时，都需要从设置表中获取数据，初始化列表控件的项目。该项任务分别又由各自的重置过程完成。

以下是这 4 个过程的详细代码解释，这些过程的流程都十分简单，不再给出过程的流程图。

```
'该过程完成对资产编号页的初始化工作，它在多页控件页单击事件中被调用，为保证该过程仅在资产编号页第一次被显示时执行，使用了一个局部静态变量以便于计算该过程当前被运行的次数
Sub 初始化资产编号页()
```

```
Static runCount As Integer
runCount = runCount + 1
'当改过程被执行超过 1 次时，将会直接跳出该过程
If runCount > 1 Then Exit Sub
数字长度.Text =  设置表.Cells(2, 1)                    '从设置表中读取数字长度
End Sub

'该过程用于初始化使用部门页
Sub  初始化使用部门页()
Static runCount As Integer
runCount = runCount + 1
'当该过程被执行超过一次时，终止执行该过程
If runCount > 1 Then Exit Sub
重置部门列表                                          '调用重置部门表过程，初始化部门列表项目
End Sub

Sub  初始化资产类别页()
Static runCount As Integer
runCount = runCount + 1
If runCount > 1 Then Exit Sub
重置类别列表                                          '调用重置类别列表过程，初始化类别列表项目
End Sub

Sub  初始化资产来源页()
Static runCount As Integer
runCount = runCount + 1
If runCount > 1 Then Exit Sub
重置来源列表                                          '调用重置来源列表过程，初始化来源列表项目
End Sub
```

6.7.4 重置列表过程代码设计

在上面讲述的初始化页过程中，后面 3 个过程都通过调用一个重置列表过程来完成各自页中列表控件项目的初始设置工作。这 3 个初始化列表项目的过程其流程大体类似，首先通过获取设置表中该存储信息所在列的最大行号，然后通过一个循环将这些记录添加到列表控件中。由于 3 个过程流程的相似性，以下只给出重置部门列表的过程流程图，其他两个过程读者可以参考该流程图加以理解。如图 6-27 所示的是重置部门列表的过程流程图。

以下是这些重置列表项目过程的详细代码解释：

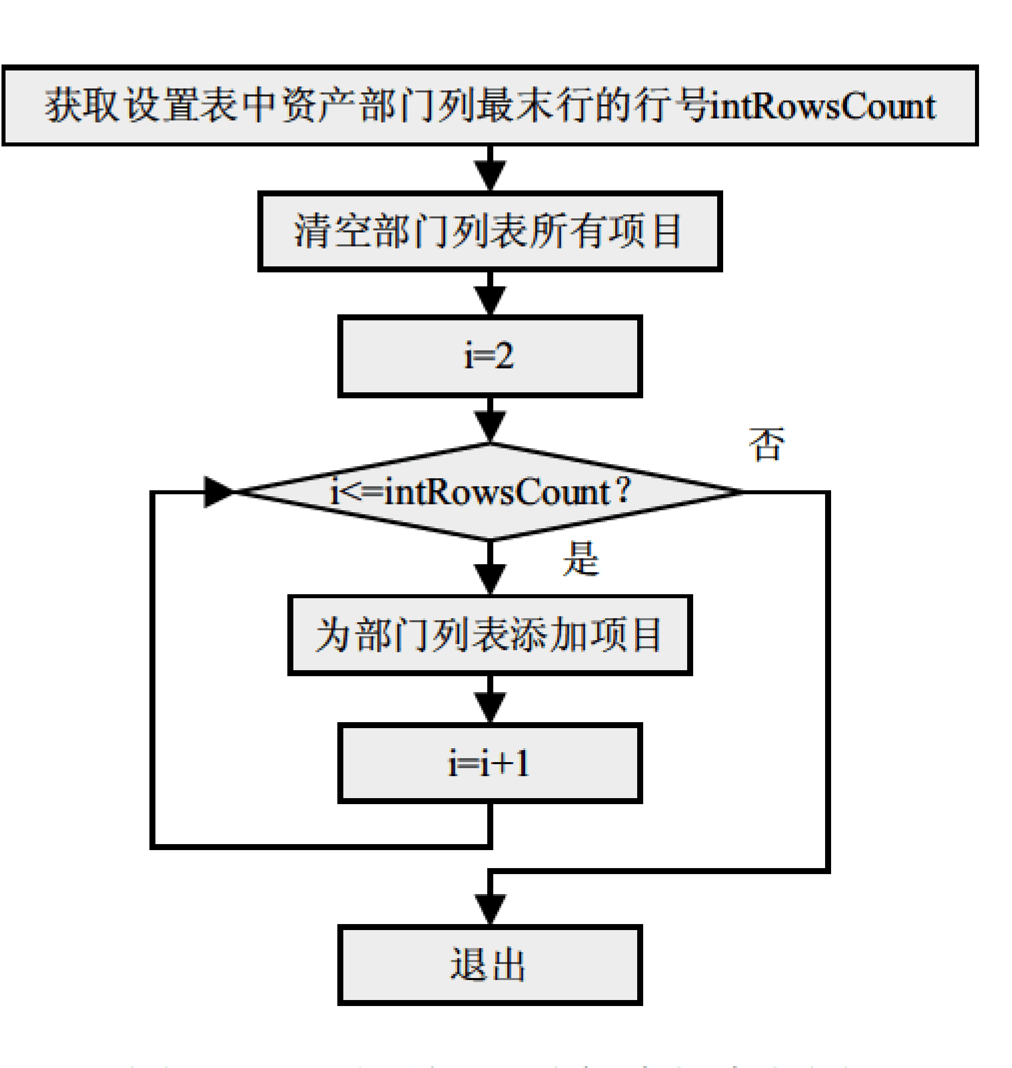


图 6-27 重置部门列表过程流程图

```
'该过程将会重置使用部门页中部门列表列表框
Sub 重置部门列表()
Dim intRowsCount As Integer, i As Integer
'获取设置表中资产部门列有记录的最尾行的行号
intRowsCount = 设置表.Range("B" & Rows.Count).End(xlUp).Row
部门列表.Clear                                          '清空部门列表所有项目
'为部门列表列表框添加记录
For i = 2 To intRowsCount                               '循环设置表资产部门列所有记录行
    部门列表.AddItem 设置表.Range("B" & i)              '为部门列表添加项目
Next
End Sub

Sub 重置类别列表()
Dim intRowsCount As Integer, i As Integer
intRowsCount = 设置表.Range("C" & Rows.Count).End(xlUp).Row '获取设置表类别列行数
类别列表.Clear                                          '清除类别列表所有项目
For i = 2 To intRowsCount                               '循环设置类别列表所有记录行
    类别列表.AddItem 设置表.Range("C" & i)              '为类别列表添加新项目
Next
End Sub

Sub 重置来源列表()
Dim intRowsCount As Integer, i As Integer
intRowsCount = 设置表.Range("E" & Rows.Count).End(xlUp).Row '获取设置表来源列行数
来源列表.Clear                                          '清除来源列表所有项目
For i = 2 To intRowsCount                               '循环设置表来源列所有记录行
    来源列表.AddItem 设置表.Range("E" & i)              '为来源列表添加新项目
Next
End Sub
```

6.7.5　多页控件单击事件代码设计

以上所介绍的 4 个页初始化过程都是在多页控件单击事件中被激发的。多页控件的页标签被单击时，该单击事件会捕获单击页的索引号，然后程序根据该索引号决定应该执行哪一个初始化页过程。以下是该事件过程的详细代码解释：

```
'当多页控件被单击时，执行以下代码，根据被单击的页的索引，有针对性地执行相应的初始化代码
'多页控件的页的序号是从 0 开始的
Private Sub MultiPage1_Click(ByVal Index As Long)
Select Case Index
    Case Is = 0
        初始化资产编号页
    Case Is = 1
        初始化使用部门页
    Case Is = 2
        初始化资产类别页
    Case Is = 3
        初始化资产来源页
```

```
End Select
End Sub
```

6.7.6 使用部门页控件单击事件代码设计

在部门页中一共有 4 个控件包含了单击事件，这些控件分部是部门列表、删除按钮、添加按钮和修改按钮。部门列表的单击事件将用户选择的项目值输入到【部门名称】文本框中，以便修改该名称。删除按钮单击事件将选择部门名称删除。添加按钮将部门名称文本框中的新部门添加到设置表。修改按钮将修改用户选择的部门名称为新名称。

在这 4 个控件的单击事件中，3 个按钮的单击事件稍微比较复杂。以下分别以文字和图示对这 3 个事件过程的流程加以介绍。

- 删除按钮单击事件：单击【删除】按钮后，程序首先获取了设置表中使用部门列末条记录所在行号，然后依次检测该列中所有部门，当有部门与用户需删除的部门相同时，程序将删除该部门信息，最后程序刷新部门列表框的数据显示，以同步用户的删除操作，然后退出整个过程。该单击事件过程的流程图如图 6-28 所示。
- 添加按钮单击事件：单击【添加】按钮后，程序首先获取设置表中使用部门列末条记录所在行号，然后依次检测该列中所有部门，当有部门与需添加的部门相同时，程序将提示部门名已存在，并直接退出过程。当检测完所有的记录行后仍然没有相同记录时，程序将把该部门保存在部门列最后一行单元格的下一个单元格，最后程序刷新部门列表框的显示，以同步用户的添加部门操作。如图 6-29 所示的是该单击事件过程的流程图。

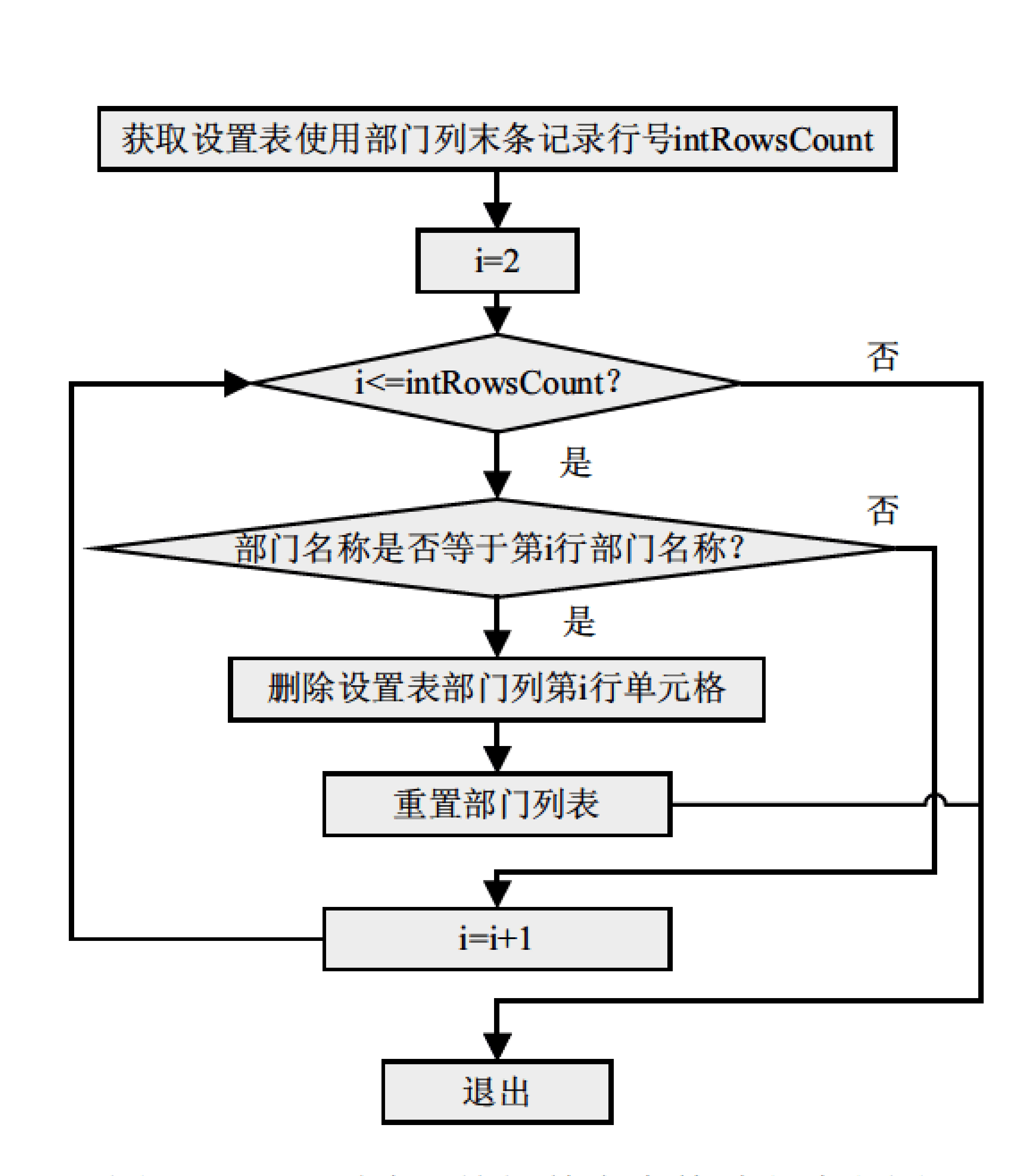


图 6-28 【删除】按钮单击事件过程流程图

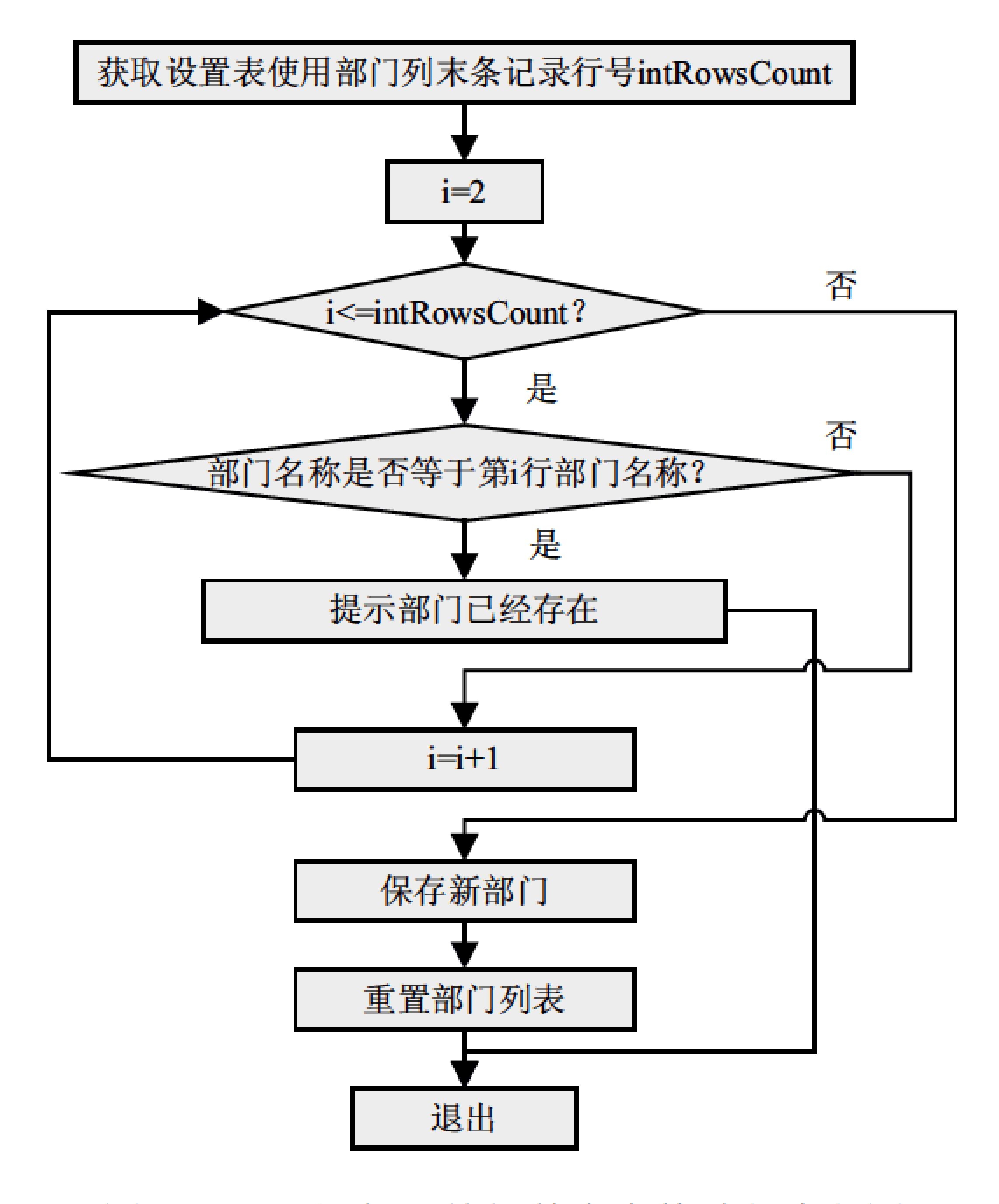


图 6-29 【添加】按钮单击事件过程流程图

❑ 修改按钮单击事件：单击【修改】按钮后，程序首先获取设置表中使用部门列末条记录所在行号，然后依次检测该列的所有部门记录。如果有部门与部门列表选择项相同，程序将把该部门的名称修改为部门名称文本框中的值。该过程的流程图如图 6-30 所示。

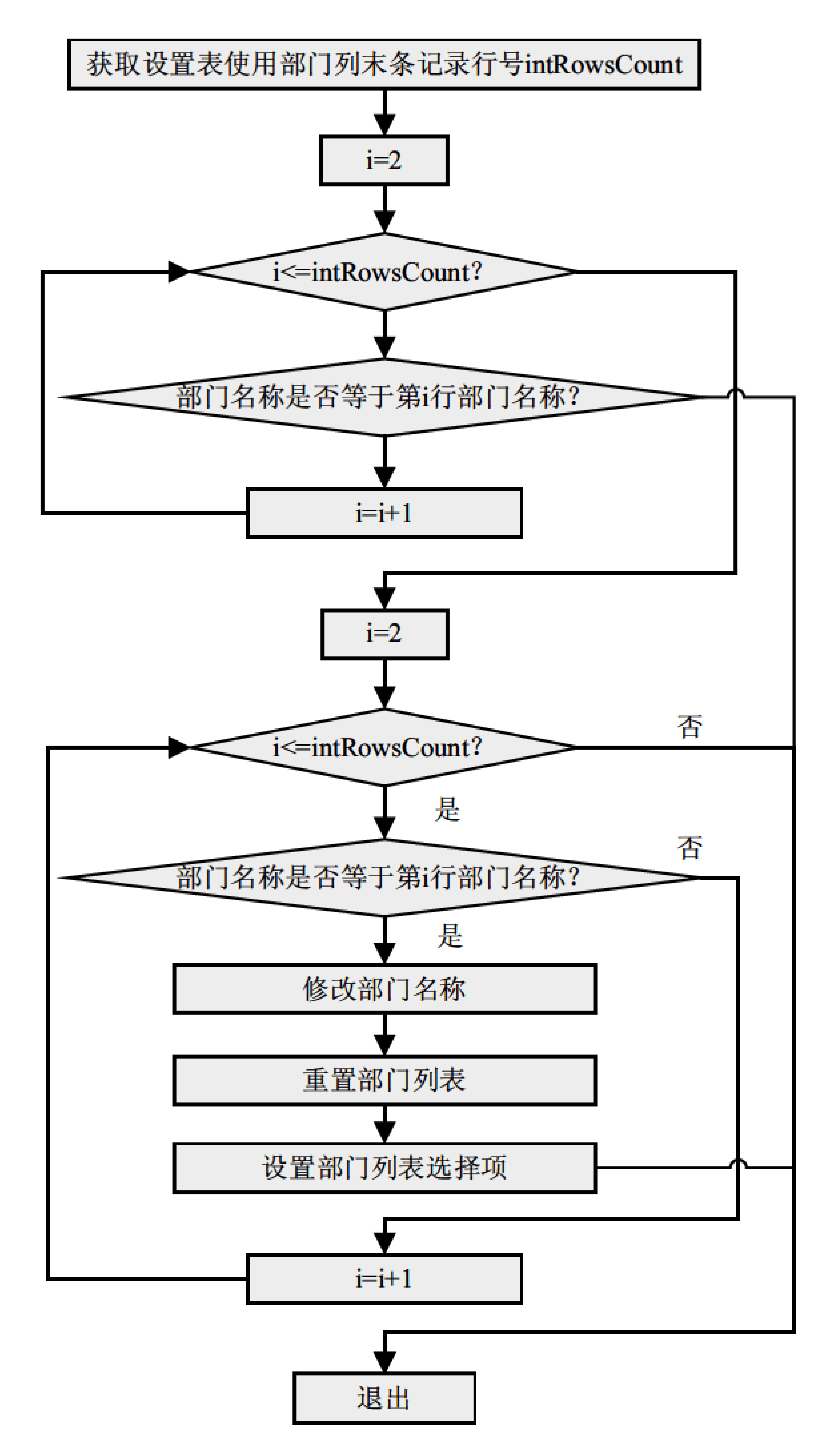


图 6-30　【修改】按钮单击事件过程流程图

以下代码是该多页控件中各个按钮的执行代码：

```
Private Sub 部门列表_Click()
部门名称.Text = 部门列表.Text                    '将用户选择的部门项目填写到部门名称文本框
End Sub

Private Sub 部门删除_Click()
Dim intRowsCount As Integer, i As Integer
intRowsCount = 设置表.Range("B" & Rows.Count).End(xlUp).Row  '获取设置表部门列末行记录行号
```

```
For i = 2 To intRowsCount                                  '循环部门列所有记录
    If 设置表.Range("B" & i) = 部门名称.Text Then            '检测第 i 个单元格是否为删除单元格
        设置表.Range("B" & i).Delete xlShiftUp               '删除第 i 个部门单元格
        重置部门列表                                          '刷新部门列表框项目
        Exit Sub                                              '退出过程
    End If
Next
End Sub

Private Sub 部门添加_Click()
Dim intRowsCount As Integer, i As Integer
intRowsCount = 设置表.Range("B" & Rows.Count).End(xlUp).Row  '获取设置表部门列末行记录行号
For i = 2 To intRowsCount                                  '循环部门列所有记录
    If 设置表.Range("B" & i) = 部门名称.Text Then            '检测第 i 个单元格是否与添加部门重复
        MsgBox "该部门已经存在！", vbOKOnly + vbInformation, "部门重复"
        Exit Sub                                              '退出过程
    End If
Next
设置表.Range("B" & (intRowsCount) + 1) = 部门名称.Text       '添加新部门
重置部门列表                                                  '刷新部门列表框项目
IsConfigChange = True                                         '标记工作簿需要保存
End Sub

Private Sub 部门修改_Click()
Dim intRowsCount As Integer, i As Integer
intRowsCount = 设置表.Range("B" & Rows.Count).End(xlUp).Row  '获取设置表部门列末行记录行号
For i = 2 To intRowsCount                                  '循环部门列所有记录
    If 设置表.Range("B" & i) = 部门名称.Text Then            '检测修改部门名是否有重复
        MsgBox "该部门已经存在！", vbOKOnly + vbInformation, "部门重复"
        Exit Sub
    End If
Next
For i = 2 To intRowsCount                                  '循环部门列所有记录
    If 设置表.Range("B" & i) = 部门列表.Text Then            '检测第 i 个单元格是否为需修改单元格
        设置表.Range("B" & i) = 部门名称.Text                 '修改第 i 个部门为新部门
        IsConfigChange = True                                 '标记工作簿需要保存
        重置部门列表                                          '刷新部门列表框项目
        部门列表.ListIndex = i - 2                            '设置部门列表被选定项目
        Exit Sub                                              '退出过程
    End If
Next
End Sub
```

6.7.7 资产类别页事件代码设计

资产类别页中包含的控件被建立时，是直接通过复制使用部门页实现的。该页所包含的功能相似性很大，其代码与使用部门页的代码相差也很小。下面不再对该页的事件代码做具体介绍。以下是该页中包含的代码：

```
Private Sub 类别列表_Click()
类别名称.Text = 类别列表.Text                              '将用户选择的资产类别填写到类别名称文本框
End Sub

Private Sub 类别删除_Click()
Dim intRowsCount As Integer, i As Integer
intRowsCount = 设置表.Range("C" & Rows.Count).End(xlUp).Row  '获取设置表类别列末行记录行号
For i = 2 To intRowsCount                                   '循环类别列所有记录
    If 设置表.Range("C" & i) = 类别名称.Text Then           '检测第 i 个类别名称是否为需删除类别
        设置表.Range("C" & i).Delete xlShiftUp              '删除第 i 个类别
        重置类别列表                                        '刷新类别列表
        Exit Sub                                            '退出过程
    End If
Next
End Sub

Private Sub 类别添加_Click()
Dim intRowsCount As Integer, i As Integer
intRowsCount = 设置表.Range("C" & Rows.Count).End(xlUp).Row  '获取设置表类别列末行记录行号
For i = 2 To intRowsCount                                   '循环类别列所有记录
    If 设置表.Range("C" & i) = 类别名称.Text Then           '检测第 i 个类别名称是否重名
        MsgBox "该类别已经存在！", vbOKOnly + vbInformation, "类别重复"
        Exit Sub
    End If
Next
设置表.Range("C" & (intRowsCount + 1)) = 类别名称.Text             '添加新类别
重置类别列表                                                       '刷新类别列表项目
IsConfigChange = True                                              '标记工作簿需保存
End Sub

Private Sub 类别修改_Click()
Dim intRowsCount As Integer, i As Integer
intRowsCount = 设置表.Range("C" & Rows.Count).End(xlUp).Row  '获取设置表类别列末行记录行号
For i = 2 To intRowsCount                                   '循环类别列表所有记录
    If 设置表.Range("C" & i) = 类别列表.Text Then           '检测第 i 个类别名称是否需修改
        设置表.Range("C" & i) = 类别名称.Text               '修改第 i 个类别的名称
        IsConfigChange = True                               '设置工作簿需要保存
        重置类别列表                                        '刷新类别列表项目
        类别列表.ListIndex = i - 2                          '设置类别列表被选定项目
        Exit Sub                                            '退出过程
    End If
Next
End Sub
```

6.7.8　资产来源页事件代码设计

资源来源页中包含的事件代码和使用部门、资产类别页的代码十分相似。以下不再针对各个事件代码加以详细介绍。各个事件的流程可以参照使用部门页中的类似事件加以理解。

```
Private Sub 来源列表_Click()
来源名称.Text = 来源列表.Text                              '将用户选择的资产来源填写到来源名称文本框
End Sub

Private Sub 来源删除_Click()
Dim intRowsCount As Integer, i As Integer
intRowsCount = 设置表.Range("E" & Rows.Count).End(xlUp).Row  '获取设置表来源列末行记录行号
For i = 2 To intRowsCount                                  '循环来源列表所有记录
    If 设置表.Range("E" & i) = 来源名称.Text Then           '检测第 i 个来源名称是否为需删除来源
        设置表.Range("E" & i).Delete xlShiftUp              '删除第 i 个来源
        重置来源列表                                        '刷新来源列表
        Exit Sub                                            '退出过程
    End If
Next
End Sub

Private Sub 来源添加_Click()
Dim intRowsCount As Integer, i As Integer
intRowsCount = 设置表.Range("E" & Rows.Count).End(xlUp).Row  '获取设置表来源列末行记录行号
For i = 2 To intRowsCount                                  '循环来源列表所有记录
    If 设置表.Range("E" & i) = 来源名称.Text Then           '检测第 i 个来源名称是否与添加来源重名
        MsgBox "该来源已经存在！", vbOKOnly + vbInformation, "来源重复"
        Exit Sub                                            '退出过程
    End If
Next
设置表.Range("E" & (intRowsCount + 1)) = 来源名称.Text            '添加新来源
重置来源列表                                                      '刷新来源列表项目
IsConfigChange = True                                             '标记工作簿需要保存
End Sub

Private Sub 来源修改_Click()
Dim intRowsCount As Integer, i As Integer
intRowsCount = 设置表.Range("E" & Rows.Count).End(xlUp).Row  '获取设置表来源列末行记录行号
For i = 2 To intRowsCount                                  '循环来源列表所有记录
    If 设置表.Range("E" & i) = 来源名称.Text Then           '检测第 i 个来源是否与修改后来源重名
        MsgBox "该来源已经存在！", vbOKOnly + vbInformation, "来源重复"
        Exit Sub                                            '退出过程
    End If
Next
For i = 2 To intRowsCount                                  '循环来源列表所有记录
    If 设置表.Range("E" & i) = 来源列表.Text Then           '检测第 i 个来源是否需要修改
        设置表.Range("E" & i) = 来源名称.Text               '修改来源名称
        IsConfigChange = True                               '标记工作簿需要保存
        重置来源列表                                        '刷新来源列表项目
        来源列表.ListIndex = i - 2                          '设置来源列表需要
        Exit Sub                                            '退出过程
    End If
Next
End Sub
```

6.8　计提日期窗体设计

计提日期窗体用于获取当前需要计提折旧的日期，并将该日期保存在设置表中。在计提折旧时，正是使用折旧日期与固定资产的投入使用日期比较判断是否需要折旧。当然所提供日期所在月份已计提过折旧的情况下，当月是不会再重新计提折旧的。

6.8.1　窗体界面设计

该窗体的界面如图 6-31 所示。由于仅仅只需要获取计提折旧的日期，因此界面比较简单，主要包含有直接作用的功能控件如表 6-4 所示。

表 6-4　计提日期窗体控件表

控　件　名	控 件 说 明
Cal 计提日期	日历控件。该控件用于获取折旧日期，在选择日期时，需要具体到日期
确定	按钮。确认当前计提折旧日期、保存，然后开始计提折旧工作
取消	按钮。退出计提折旧日期窗口

在默认情况下，窗体设计的工具栏中并不包含日历控件。通过以下步骤的操作可以通过工具栏直接使用日历控件。首先依次选择【视图】|【工具箱】命令。然后在工具箱空白处右击，在弹出的快捷菜单中选择【附加控件】命令。在弹出的【附加控件】对话框中的【可用控件】列表框中找到日历控件，选中并确定即可（如图 6-32 所示）。



图 6-31　计提日期界面



图 6-32　附加控件

6.8.2　窗体代码设计

该窗体的代码完成的主要工作包括获取正确的计提折旧日期、保存计提折旧日期、调用计提折旧公共过程。这些工作通过该窗体的相应日历控件和按钮控件的事件过程来激发。

在该窗体包含了 3 个事件代码，分别是窗口初始化事件、【确定】按钮单击事件和【取消】按钮单击事件。

窗口初始化时，程序将当前日期显示在计提日期日历控件上。单击【确定】按钮时，程序检测用户是否输入了正确的日期，然后分别执行相应的操作。【取消】按钮被单击时直接退出窗口。3 个过程中【确定】按钮的单击事件较复杂，以下只给出该过程的流程图，如图 6-33 所示。

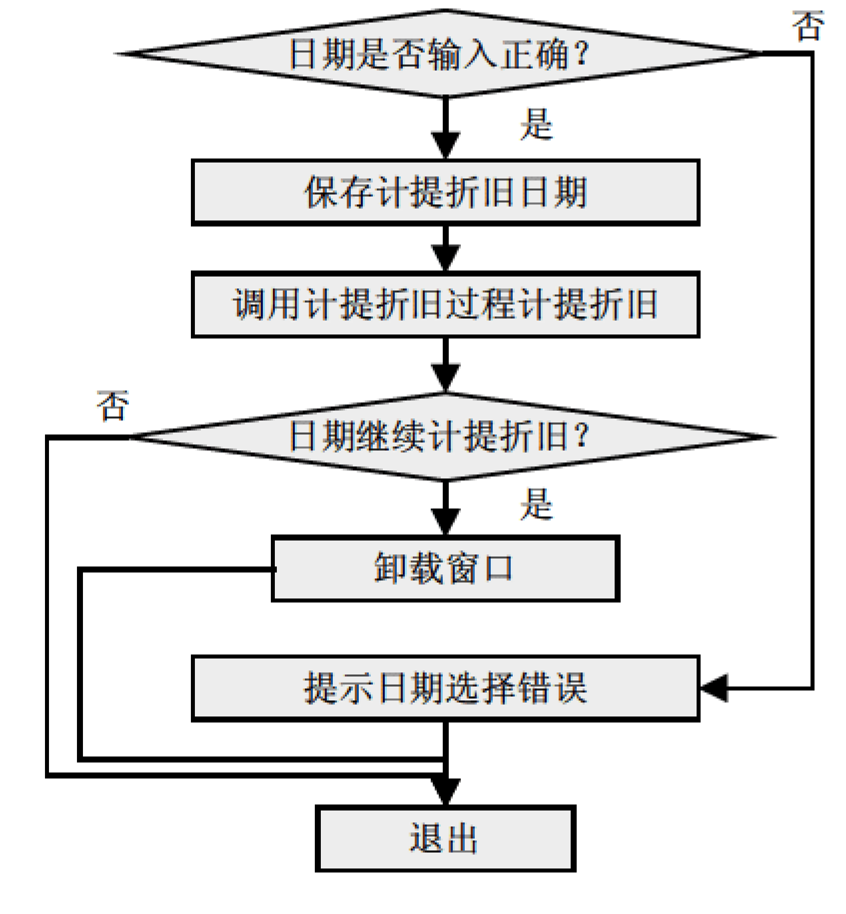


图 6-33 【确定】按钮单击事件过程流程图

以下是该窗体的具体代码解释：

```
'初始化窗体，将该窗体首次显示的日期设置为设置表中保存的日期
Private Sub UserForm_Initialize()
Cal 计提日期.Value = Format(Now, "YYYY-MM-DD")            '设置计提日期控件显示日期
End Sub

'卸载窗体
Private Sub 取消_Click()
Unload Me                                                  '卸载窗口
End Sub

'保存计提折旧日期并调用计提折旧过程
Private Sub 确定_Click()
'日历控件中如果没有选择日期，将会被确认为没有获取日期，通过检测日历控件的 Day 属性确认正确
 输入日期
If 计提日期.Day Then                                       '检测计提日期是否设置正确
    设置表.Range("D2") = Cal 计提日期.Value                '保存计提折旧日期
    计提折旧                                               '开始计提折旧
    'If 结构用于确认用户是否继续计提折旧，单击确定按钮时，不退出窗体，可以继续选择其他计提日
     期继续计提折旧
    If MsgBox("计提折旧完成，是否继续", vbOKCancel + vbInformation) = vbCancel Then
        Unload Me                                          '卸载窗体
    End If
Else
    '提示计提日期选择不正确
    MsgBox "请选中一个日期！", vbOKOnly + vbInformation    '提示计提日期选择不正确
End If
End Sub
```

6.9 进度窗体设计

在固定资产登记操作中，当固定资产所有信息项目都输入后选择保存时，程序需要将该固定资产的信息写入对应该固定资产的一个单独固定资产折旧明细表中，还要将该条固定资

产信息写入固定资产登记表中。该操作中数据读写操作很频繁，另外还涉及到新建表对象操作，可能造成程序暂时的无反应现象。为了避免让用户误认为程序挂起，程序使用了进度条窗口，以便于提示保存固定资产信息工作的进度状况。

由于该窗体的界面仅包含了一个进度条控件，窗体的界面设计比较简单，本节将不分开讲述界面设计。在默认的窗体设计工具箱中，并没有进度条，要调用该控件，首先要引用 Microsoft Windows Common Controls 6.0，选择【工具】|【引用】命令，然后在【引用】窗口中找到该控件，单击【确定】按钮即可；接着在工具箱上右击，在弹出的快捷菜单中选择【附加控件】命令，选择 Microsoft ProgressBar Control,Version6.0 之后，即可在工具箱中直接选择进度条控件。该窗体的界面如图 6-34 所示。

该窗体仅包含一个窗体激活事件过程。在该窗体的事件代码中，有两个循环，分别完成向固定资产明细表写入数据和向固定资产统计表写入数据。两个循环写入的数据都一样，因而在计算进度时，将整个进度划分成两块，各占 50%。例如计算写入资产明细表时，进度条的值为 Int(i / UBound(myArray) * 50)。i / 2*UBound(myArray)得到的是当前循环的该次循环在整个进度中占的比例。因为进度条控件的值不需要百分号，所以需要乘以 100，然后对该值取整。该过程的执行流程图如图 6-35 所示。

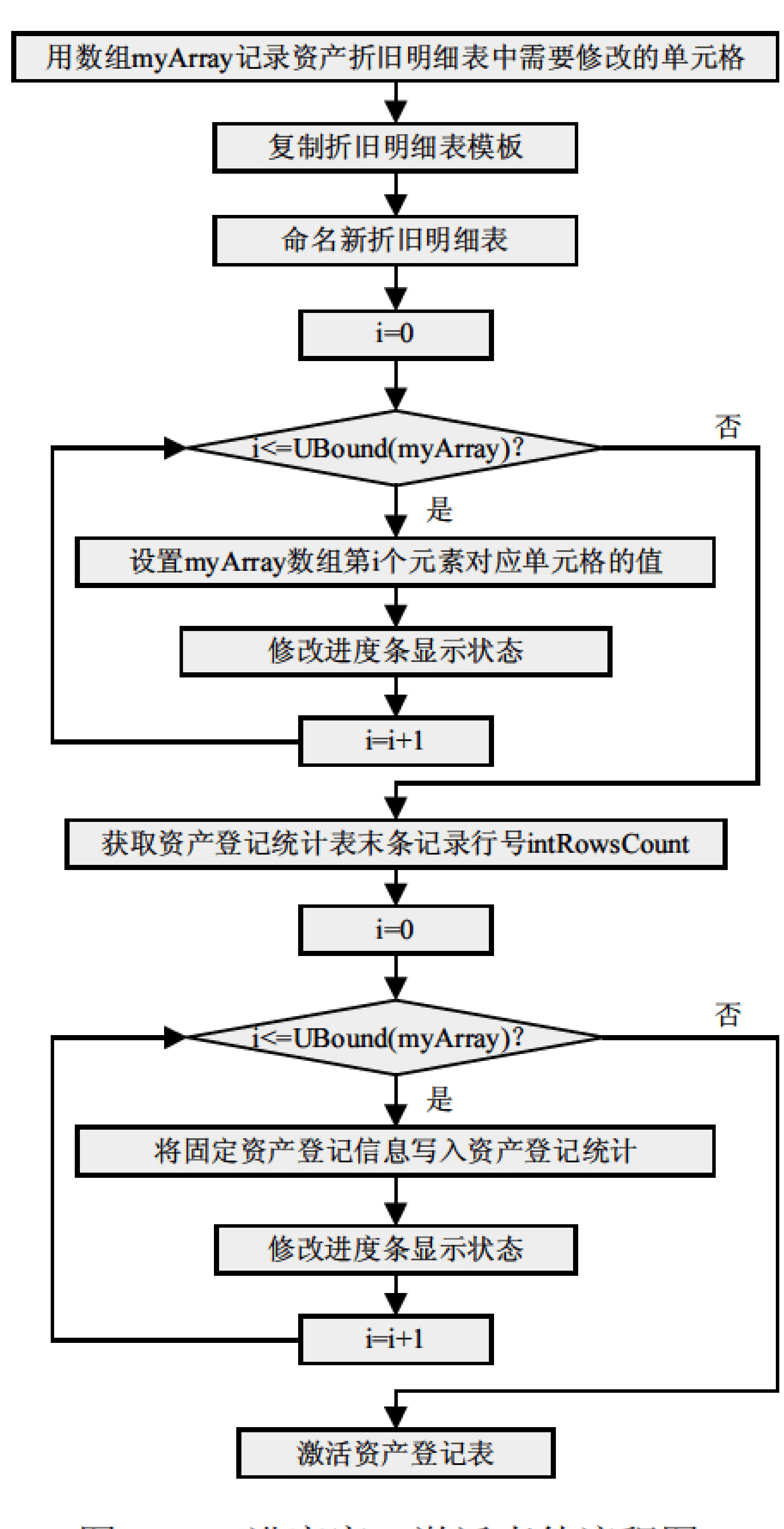


图 6-34　固定资产登记保存进度

图 6-35　进度窗口激活事件流程图

该过程的详细代码解释如下：

```
Private Sub UserForm_Activate()
Dim ws As Worksheet, myArray() As Variant, i As Integer
Dim intRowsCount As Integer
'用数组记录固定资产折旧明细表中需要修改的单元格，以便于循环操作
myArray = Array("K1", "B3", "B4", "B5", "B6", "F3", "F4", "F5", "F6", _
                "J3", "J4", "J5", "J6", "L3", "L4", "L5", "L6")
'关闭工作簿刷新
Application.ScreenUpdating = False
'复制单项固定折旧明细模板表，新产生的工作表位于该表的前面
单项资产折旧明细模板.Copy Before:=Worksheets("单项固定资产折旧明细模板")
With ActiveSheet
    '新的工作表的标签名修改为"MX-" +资产编号+ "-"资产名称的格式
    .Name = "MX-" & 资产登记表.Range("B3") & "-" & 资产登记表.Range("B4")
    For i = 0 To UBound(myArray)
        '向资产明细表写入数据
        .Range(myArray(i)).Formula = 资产登记表.Range(myArray(i)).Formula
        '修改进度条显示状态
        ProgressBar1.Value = Int(i / UBound(myArray) * 50)
    Next
End With
intRowsCount = 资产登记统计.Range("B" & Rows.Count).End(xlUp).Row
For i = 0 To UBound(myArray)
    '将固定资产登记信息写入资产登记统计
    资产登记统计.Cells(intRowsCount + 1, i + 2) = ActiveSheet.Range(myArray(i))
    '修改进度条显示状态，因为前面已经完成了50%的工作，所以该值需要加上50
    ProgressBar1.Value = 50 + Int(i / UBound(myArray) * 50)
Next
'重新激活资产登记表，开启工作簿刷新并卸载窗体
资产登记表.Activate
Application.ScreenUpdating = True
Unload Me
End Sub
```

6.10 利用数据窗体设计

利用数据窗体用于显示在固定资产登记统计表中已经登记过的固定资产的详细信息。该窗体被固定资产登记表中的利用数据按钮所调用。通过在该窗体中单击按钮，当前被选择的固定资产的数据会自动写入到固定资产登记表中。如果当前需要登记的固定资产的各个数据和当前选择的固定资产类似，可以使用该方法迅速建立资料。



图 6-36 利用数据界面

6.10.1 窗体界面设计

该窗体包含的控件数量少，此处不再列出表格一一介绍。窗体的界面效果如图 6-36 所示，包含的控件有：

- ListView 控件。显示在固定资产登记统计表中已经登记过的固定资产的详细信息。
- 【确定】按钮。单击该按钮后，当前选择的固定资产的信息会写入资产登记表中，类似于在该项固定资产上双击。
- 【取消】按钮。单击该按钮后，将会退出该窗体。

6.10.2　窗体初始化代码设计

在窗体的初始化事件中，主要完成的是 ListView 控件的显示设置。要完成该控件的显示设置，程序将该工作分为 3 个步骤。首先程序设置了该控件部分显示属性，然后初始化了该控件的列头，最后为该控件添加显示项目。

控件列头数据的来源是资产登记统计表第一行的数据。程序通过一个 For 循环将这些数据依次写入控件的列头中。控件中所有项目的数据来源是资产登记统计表中登记的固定资产数据。添加项目时，程序通过一个嵌套 For 循环依次为控件添加新项目，然后再为新项目添加子项。如图 6-37 所示的是窗口初始化的过程流程图。

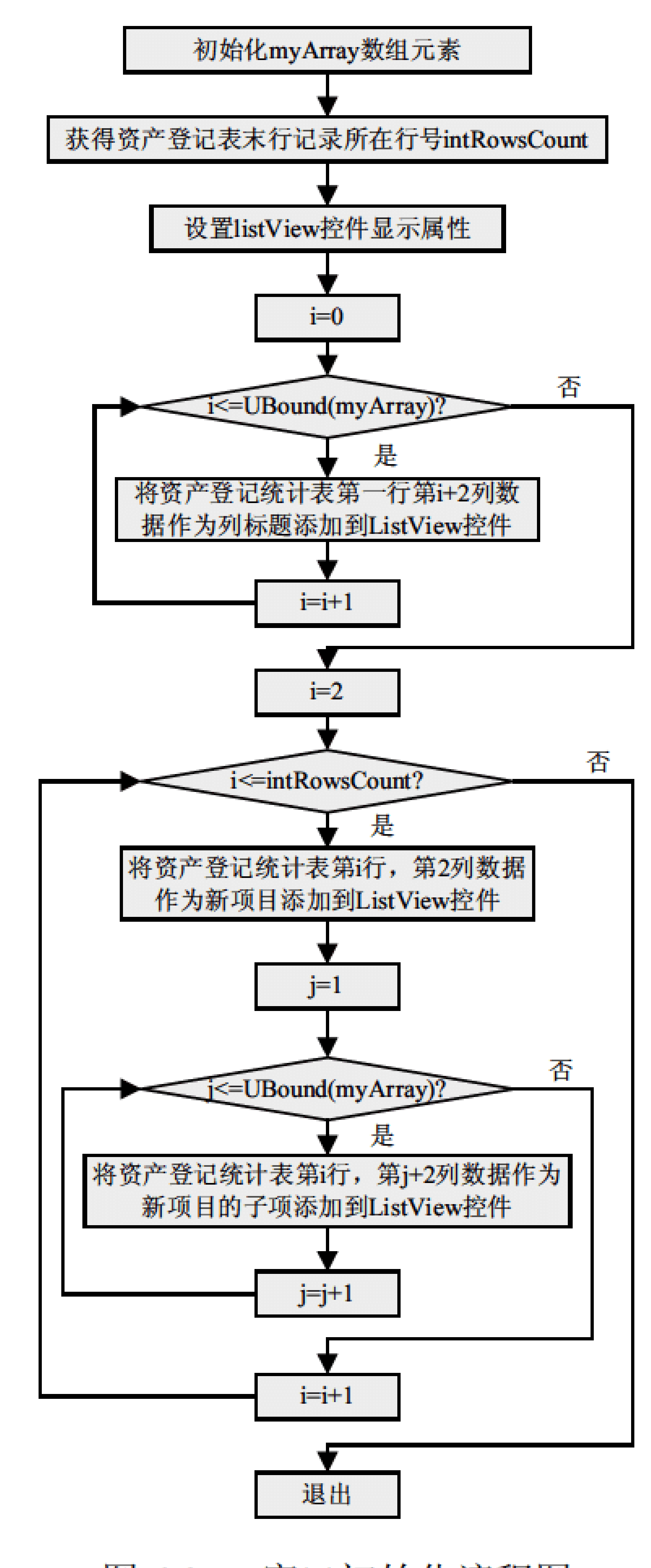


图 6-37　窗口初始化流程图

以下是该窗体初始化过程的详细代码解释：

```
'该数组记录需要读写的各个单元格的位置，以便于对这些单元格进行循环操作
Dim myArray() As Variant

窗体初始化代码
Private Sub UserForm_Initialize()
Dim i As Integer, intRowsCount As Integer
Dim 列表项 As ListItem, j As Integer
'将需要读写操作的单元格位置写入数组中
myArray = Array("K1", "B3", "B4", "B5", "B6", "F3", "F4", "F5", "F6", _
            "J3", "J4", "J5", "J6", "L3", "L4", "L5", "L6")
'获得资产登记表中最后一项固定资产所在的行号
intRowsCount = 资产登记统计.Range("B" & Rows.Count).End(xlUp).Row
With ListView1
    .ColumnHeaders.Clear                '清除列头
    .ListItems.Clear                    '清除所有项目
    .View = lvwReport                   '设置显示模式为 lvwReport 报告形式
    .FullRowSelect = True               '设置整行选择 FullRowSelect 属性
    .Gridlines = True                   '设置显示网格线
    .Sorted = True                      '设置排列
    .HideSelection = False              '窗体失去焦点时，被选择的项目以灰色背景显示
    '获得 ListView 控件的所有列头，这些数据位于资产登记统计表中第一行，其中第一列被返回按钮
     占据
    For i = 0 To UBound(myArray)
        .ColumnHeaders.Add , , 资产登记统计.Cells(1, i + 2)          '添加新列标题
    Next
    '向 ListView 控件添加项目
    For i = 2 To intRowsCount           '循环资产登记统计表所有记录行
        '首先为 ListView 控件建立一新项目，把该项目对象赋予给一个对象变量“列表项”
        Set 列表项 = .ListItems.Add(i -1, , 资产登记统计.Cells(i, 2))     '添加新项目
        '修改该项目中的子项的内容
        With 列表项
            For j = 1 To UBound(myArray)                               '循环所有子项对应列
                .SubItems(j) = 资产登记统计.Cells(i, j + 2)             '为新项目添加子项
            Next
        End With
    Next
End With
End Sub
```

6.10.3　窗口控件事件代码设计

窗口中包含的控件数量比较少，代码也不复杂。这里将这些控件的相关代码归纳为一个小节加以介绍。窗口中有关控件的代码包括 ListView 控件的双击事件、【确定】按钮的单击事件和【取消】按钮的单击事件。这几个事件当中只有【确定】按钮的代码较多，以下将重

点介绍该事件。

当用户在 ListView 控件中双击时，程序需要将用户在 ListView 控件中选定项目的数据写入资产登记表中，而【确定】按钮所完成的任务正是该项工作，因而双击 ListView 控件时，只需要调用【确定】按钮单击事件即可。单击【取消】按钮时直接退出窗口。

单击【确定】按钮时，程序将把用户选定项目的数据输入到资产登记表中。首先程序检测了 ListView 控件中是否有项目被选定，然后程序获取选定项目的索引号，最后程序将对应该索引号的资产登记统计表的记录数据写入到资产登记表中。

写入数据时使用了一个 For 循环，该循环通过循环 myArray 定义的所有单元格，依次写入数据。但是该数组中的单元格地址还包含了使用公式的单元格，这些单元格是不需要赋值的，这些公式单元格会自动重新计算获取值，因而需要在循环中排除这些单元格的赋值操作。如图 6-38 所示的是【确定】按钮单击事件过程的流程图。

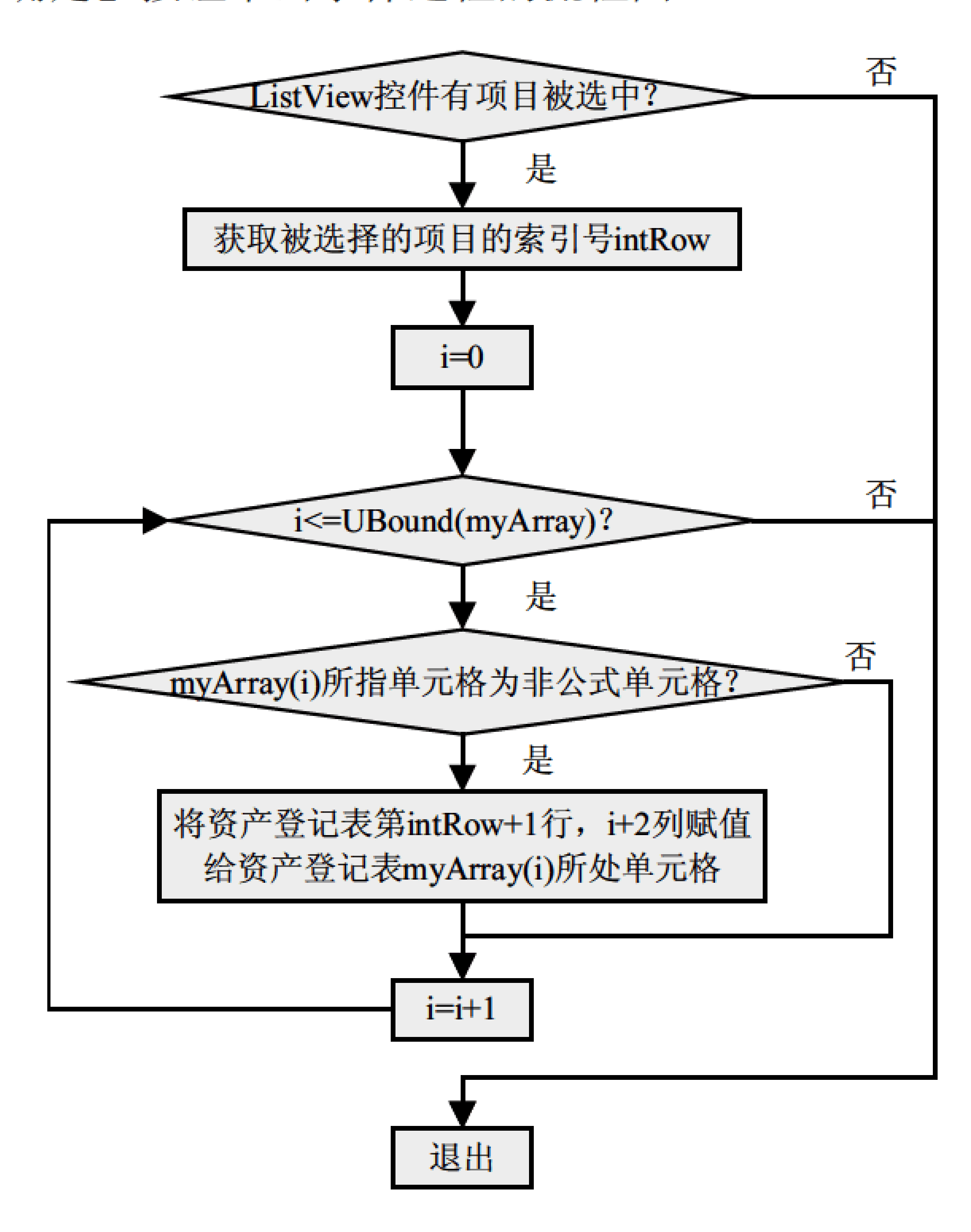


图 6-38　【确定】按钮单击事件过程的流程图

以下是该窗口中控件的事件代码详细解释：

```
'双击 ListVeiw 控件时，直接执行确定按钮单击事件过程，将数据写入表中
Private Sub ListView1_DblClick()
确定_Click                                          '调用确定按钮单击事件过程
End Sub

'取消按钮单击事件
Private Sub 取消_Click()
```

```
Unload Me                                                    '卸载窗口
End Sub

'确定按钮单击事件
Private Sub 确定_Click()
Dim i As Integer, intRow As Integer
'关闭工作簿刷新
Application.ScreenUpdating = False
'检查 ListView 控件中是否有项目被选中，当有项目被选中时，执行以下代码
If Not (ListView1.SelectedItem Is Nothing) Then
    '获取被选择的项目的索引号，该索引号从 1 开始
    intRow = ListView1.SelectedItem.Index
    For i = 0 To UBound(myArray)                             '循环所有需输入数据的单元格
    '如果写入的单元格不是公式自动产生数据，将该数据写入到资产登记表中
        If myArray(i) <> "J5" And myArray(i) <> "J6" And myArray(i) <> "L4" And myArray(i) <> _
                "L5" And myArray(i) <> "L6" Then
            资产登记表.Range(myArray(i))=资产登记统计.Cells(intRow + 1, i + 2)     '设置单元格的值
        End If
    Next
End If
Application.ScreenUpdating = True
End Sub
```

6.11 输入辅助窗体设计

输入辅助窗体在系统中一般应用到两个地方：一是在固定资产登记中，辅助输入固定资产信息；另一个是在首页中选择查看单项资产时，用于获取单项资产表的名称。在该窗体中双击需要的项或选择该项后单击【确定】按钮都可以完成选择工作。

6.11.1 窗体界面设计

该窗体的界面比较简洁，窗体的界面效果如图 6-39 所示。该窗口显示的是所有已登记的资产。

其包含以下控件：

- ❑ 框架控件：该控件用于提示显示一些信息，也将列表框控件划分出来，以达到醒目的效果。
- ❑ 列表框控件：该控件用于显示列表信息。针对不同的情况，列表框将显示不同的内容。
- ❑ 【确定】按钮：单击该按钮后，将记录该条选择信息，执行相应的过程，然后退出该窗体。

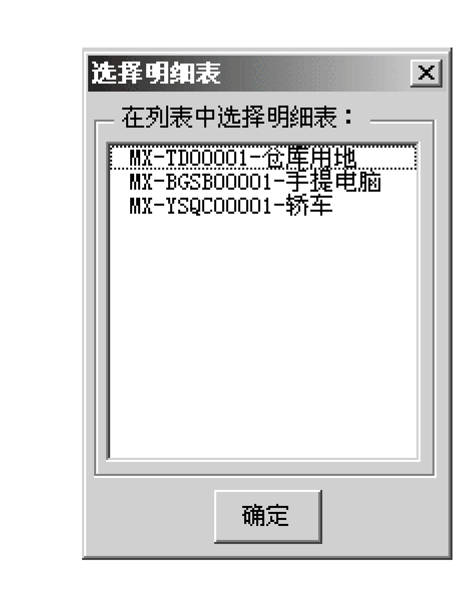


图 6-39　辅助输入窗口界面

6.11.2　窗体初始化与卸载事件代码设计

该窗口作为一个辅助输入窗口，需要根据用户的选择情况来完成窗口中列表框项目的初始化工作，因而在窗口初始化时，过程稍微比较复杂，下面将对该事件的流程进行详细介绍。窗口卸载时，如果用户没有选择任何项目，需要置空存储选择结果的公共变量，同时还需要设置变量“Is 选择明细表”为假。

该窗口还会在首页单击【查看单项资产】按钮时显示。在这种情况下显示窗口的设置与其他情况不具有相似性。因而，需要根据“Is 选择明细表”公共变量来区别不同情况的显示操作。

当窗口被初始化时，程序首先根据“Is 选择明细表”公共变量确认用户是否在首页单击了【查看单项资产】按钮。当单击该按钮时，程序修改了窗口与窗口中框架的 Caption 属性，并且依次检测工作簿中所有工作表，将属于折旧明细的工作表名称添加到窗口列表框中。当用户在输入资产信息时双击单元格，程序首先修改窗口与窗口中框架的 Caption 属性，然后从设置表中读取数据填充到列表框中。

窗口激活事件过程的代码较多，而且其中使用了一个 Select Case 分支结构。以下为了说明的方便，将该过程的流程分为 3 个流程图加以说明，如图 6-40~图 6-42 所示。其中第一个流程图为主过程流程图，第二个流程图说明选择明细表时的初始化操作，第三个流程图显示辅助输入资产类别时的初始化操作。辅助输入时不同情况的程序流程大体一致，因而这里只说明辅助输入资产类别的流程。其他情况读者可以参考该流程加以理解。

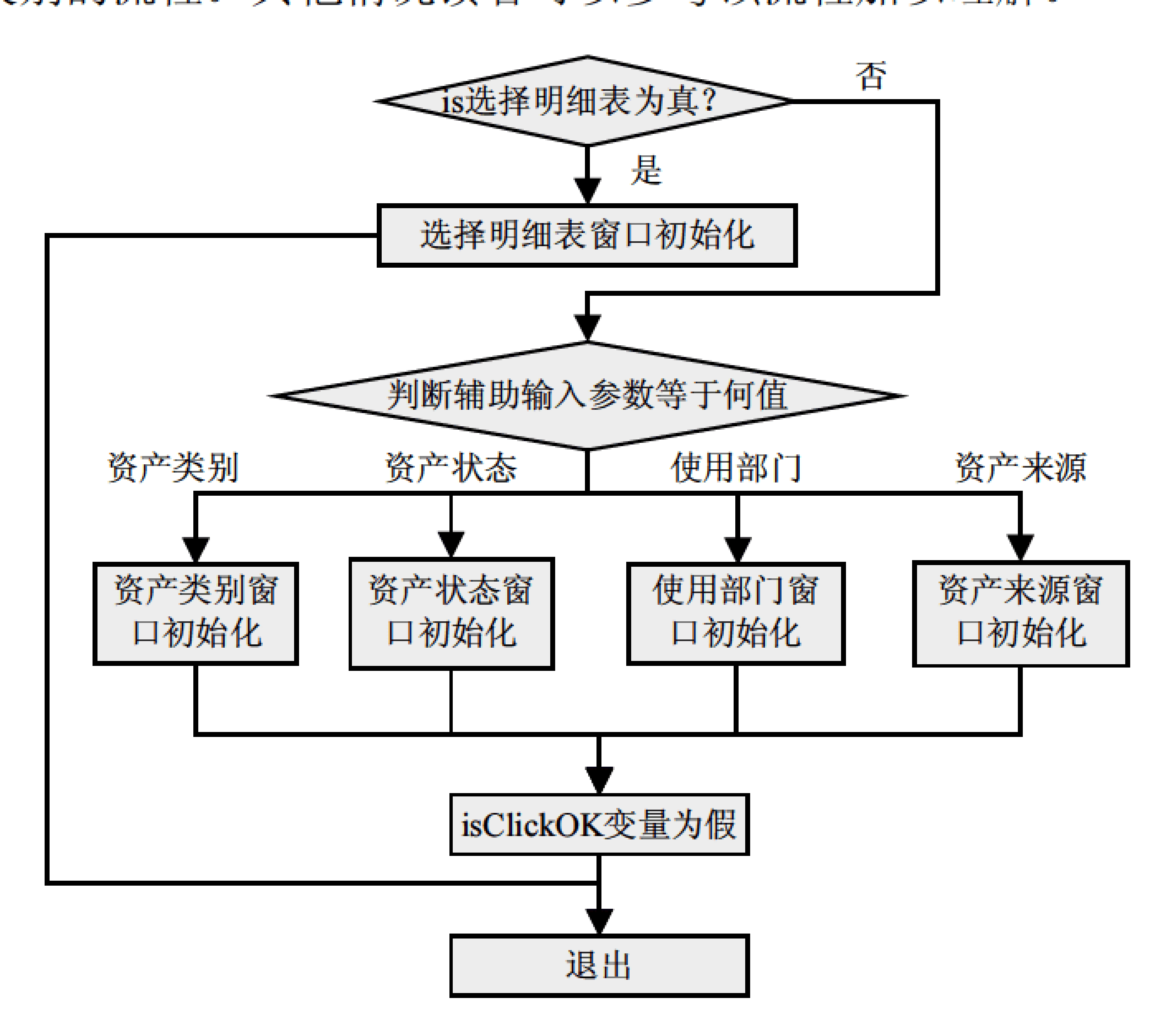


图 6-40　窗口初始化流程图

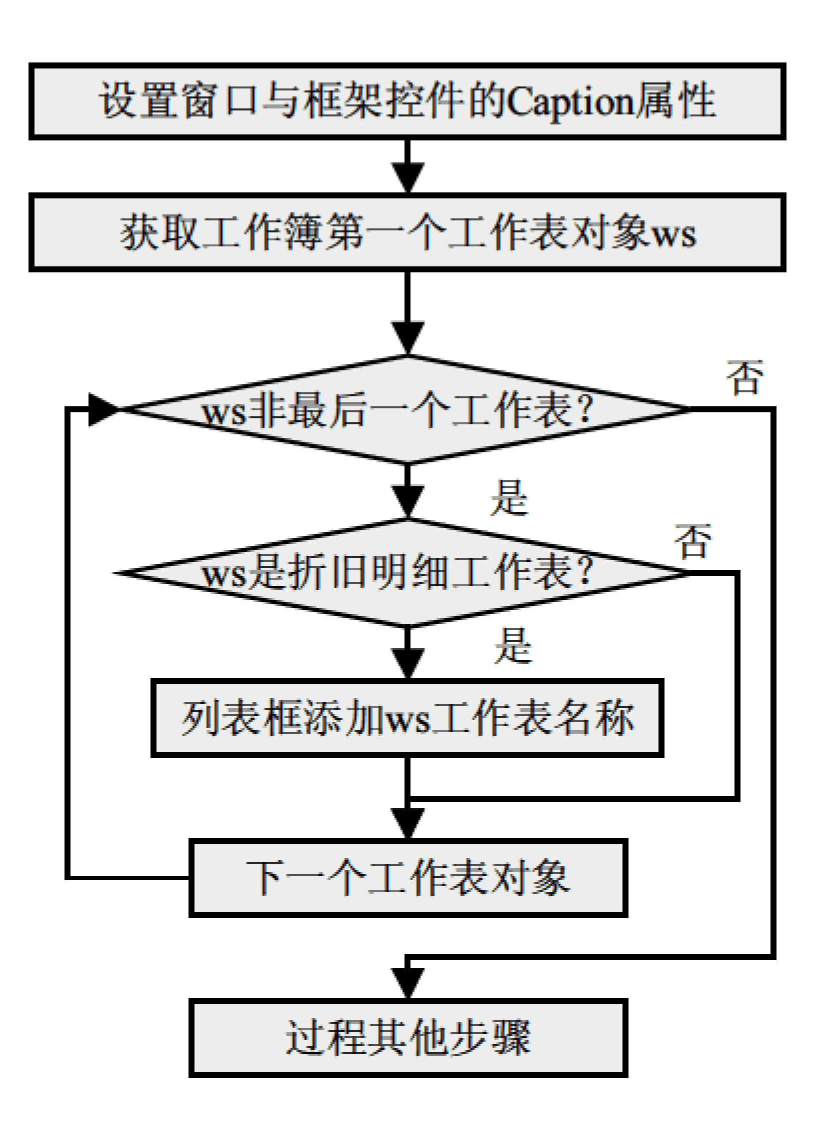


图 6-41　选择明细表窗口初始化流程图

设置窗口与框架控件的Caption属性
获取设置表资产类别列末行记录行号intRowsCount
清空内容列表
i=2
i<=intRowsCount?
否
是
列表框添加资产类别项
下一个工作表对象
过程其他步骤

图 6-42　输入资产类别窗口初始化流程图

以下是该窗口的初始化与卸载事件过程详细代码解释：

```
Dim IsClickOK As Boolean

Private Sub UserForm_Initialize()
Dim i As Integer, intRowsCount As Integer
Dim ws As Worksheet
If Is 选择明细表 Then
    Me.Caption = "选择明细表"                              '设置窗口 Caption 属性
    frame 外框.Caption = "在列表中选择明细表："               '设置框架 Caption 属性
    For Each ws In ThisWorkbook.Worksheets                 '循环工作簿中所有工作表
        If Left(ws.Name, 2) = "MX" Then                    '检测工作表是否为折旧明细工作表
            内容列表.AddItem ws.Name                        '列表框添加工作表名称
        End If
    Next
Else
    Select Case 辅助窗口参数
        Case Is = "资产类别"                                '检测辅助窗口参数是否为资产类别
            Me.Caption = "资产类别"                         '设置窗口 Caption 属性
            frame 外框.Caption = "在列表中选择资产类别："     '设置框架 Caption 属性
            '获取设置表资产类别列末条记录行号
            intRowsCount = 设置表.Range("C" & Rows.Count).End(xlUp).Row
            内容列表.Clear                                  '清除内容列表所有项目
            For i = 2 To intRowsCount                      '循环所有资产类别记录
                内容列表.AddItem 设置表.Range("C" & i)      '为内容列表添加资产类别项目
            Next
        Case Is = "资产状态"                                '检测辅助窗口参数是否为资产状态
            Me.Caption = "资产状态"                         '设置窗口 Caption 属性
```

```
            frame 外框.Caption = "在列表中选择资产状态：" '设置框架 Caption 属性
            '获取设置表资产状态列末条记录行号
            intRowsCount = 设置表.Range("D" & Rows.Count).End(xlUp).Row
            内容列表.Clear                          '清除内容列表所有项目
            For i = 2 To intRowsCount               '循环所有资产状态记录
                内容列表.AddItem 设置表.Range("D" & i)   '为内容列表添加资产状态项目
            Next
        Case Is = "使用部门"                        '检测辅助窗口参数是否为使用部门
            Me.Caption = "使用部门"                 '设置窗口 Caption 属性
            frame 外框.Caption = "在列表中选择使用部门：" '设置框架 Caption 属性
            '获取设置表使用部门列末条记录行号
            intRowsCount = 设置表.Range("B" & Rows.Count).End(xlUp).Row
            内容列表.Clear                          '清除内容列表所有项目
            For i = 2 To intRowsCount               '循环所有使用部门记录
                内容列表.AddItem 设置表.Range("B" & i)   '为内容列表添加使用部门项目
            Next
        Case Is = "资产来源"                        '检测辅助窗口参数是否为资产来源
            Me.Caption = "资产来源"                 '设置窗口 Caption 属性
            frame 外框.Caption = "在列表中选择资产来源：" '设置框架 Caption 属性
            '获取设置表资产来源列末条记录行号
            intRowsCount = 设置表.Range("E" & Rows.Count).End(xlUp).Row
            内容列表.Clear                          '清除内容列表所有项目
            For i = 2 To intRowsCount               '循环所有资产来源记录
                内容列表.AddItem 设置表.Range("E" & i)   '为内容列表添加资产来源项目
            Next
    End Select
    IsClickOK = False                               '设置 IsClickOK 为假
End If
End Sub

Private Sub UserForm_Terminate()
If Not IsClickOK Then                               '检测是否单击了确定按钮
    辅助窗口参数 = ""                               '设置辅助输入参数为空
End If
Is 选择明细表 = False                               '设置选择明细表
End Sub
```

6.11.3　窗口控件事件代码设计

窗口中包含的控件数量不多，相关控件的代码页不多。窗口中只包含了两个事件代码，分别是列表控件双击事件和【确定】按钮单击事件。内容列表被双击时，直接调用【确定】按钮单击事件过程。在【确定】按钮单击事件过程中需要根据选择明细表变量确定接下来的操作。以下是窗口控件的事件代码详细解释：

```
Private Sub 内容列表_DblClick(ByVal Cancel As MSForms.ReturnBoolean)
确定_Click                                          '调用确定按钮单击事件过程
End Sub

Private Sub 确定_Click()
If Is 选择明细表 Then
    On Error Resume Next
    Worksheets(内容列表.Text).Select                '激活用户选择的工作表
Else
    IsClickOK = True                                '设置 isClickOK 变量为真
    辅助窗口参数 = 内容列表.Text                     '存储用户选择项目数据
End If
Unload Me
End Sub
```

6.12　公共代码模块设计

在该系统的模块中，包含两个模块公共变量模块与公共过程模块。公共变量模块存储了系统的所有公共变量的定义。公共过程模块中存储的是所有公共的过程代码。

6.12.1　公共变量模块

公共变量模块定义了所有系统需要使用的公共变量。以下代码解释了这些公共变量的用途：

```
'该变量保存在辅助窗口中选择得到的相关数据
Public 辅助窗口参数 As String
'返回按钮有两种情况，在首页中操作后跳转到其他的页面时，返回就是返回首页；在固定资产登记统计
表中页可以双击项目后跳转到资产折旧明细表，当在资产折旧明细表中单击返回按钮时需要返回固定资
产登记统计表。该变量用于定义是否返回资产登记统计表。
Public 是否返回统计表 As Boolean
'对于辅助输入窗口，当是属于在首页中选择查看单项资产折旧时，该情况不同于在资产登记表中的辅助
 输入情况，用该变量区别这两种情况。从而系统可以针对两种情况采取不同的操作
Public Is 选择明细表 As Boolean
```

6.12.2　跳转按钮宏过程代码设计

在公共模块当中包含了系统中跳转按钮的宏过程代码，这些宏过程分别完成系统中各个按钮单击时的跳转任务。宏过程的代码都比较简单，基本上都是直接打开一个窗口、激活一个工作表或者调用某个过程。以下是这些宏过程的详细代码解释：

```
Sub 基本设置()
frm 基本设置.Show                                  '显示基本设置窗口
End Sub

Sub 折旧明细表()
Is 选择明细表 = True                               '标记选择明细表变量
frm 输入辅助.Show                                  '显示输入辅助窗口
End Sub

Sub 固定资产登记()
资产登记表.Activate                                '激活资产登记表工作表
End Sub

Sub 返回()
If 是否返回统计表 Then                             '检查是否返回统计表工作表
    资产登记统计.Activate                          '激活资产登记统计表工作表
Else
    首页.Activate                                  '激活首页工作表
End If
是否返回统计表 = False                             '设置是否返回统计表
End Sub

Sub 固定资产统计()
资产登记统计.Activate                              '激活资产登记统计表
End Sub

Sub 折旧与现值统计()
资产折旧与现值统计.Activate                        '激活资产折旧与现值统计工作表
End Sub

Sub 计提日期()
frm 计提日期.Show                                  '显示计提日期窗口
End Sub

Sub 利用数据()
frm 利用数据.Show                                  '显示利用数据窗口
End Sub
```

6.12.3　资产类别拼音函数代码设计

资产类别拼音函数根据获取的资产类别字符串参数返回该参数的拼音。该过程首先将资产类别字符串的汉字逐个保存到同等长度的字符串数组中，然后逐个获取每个汉字字符的拼音头字母，最后将这些拼音头字母连接起来作为结果。该过程的流程图如图 6-43 所示。

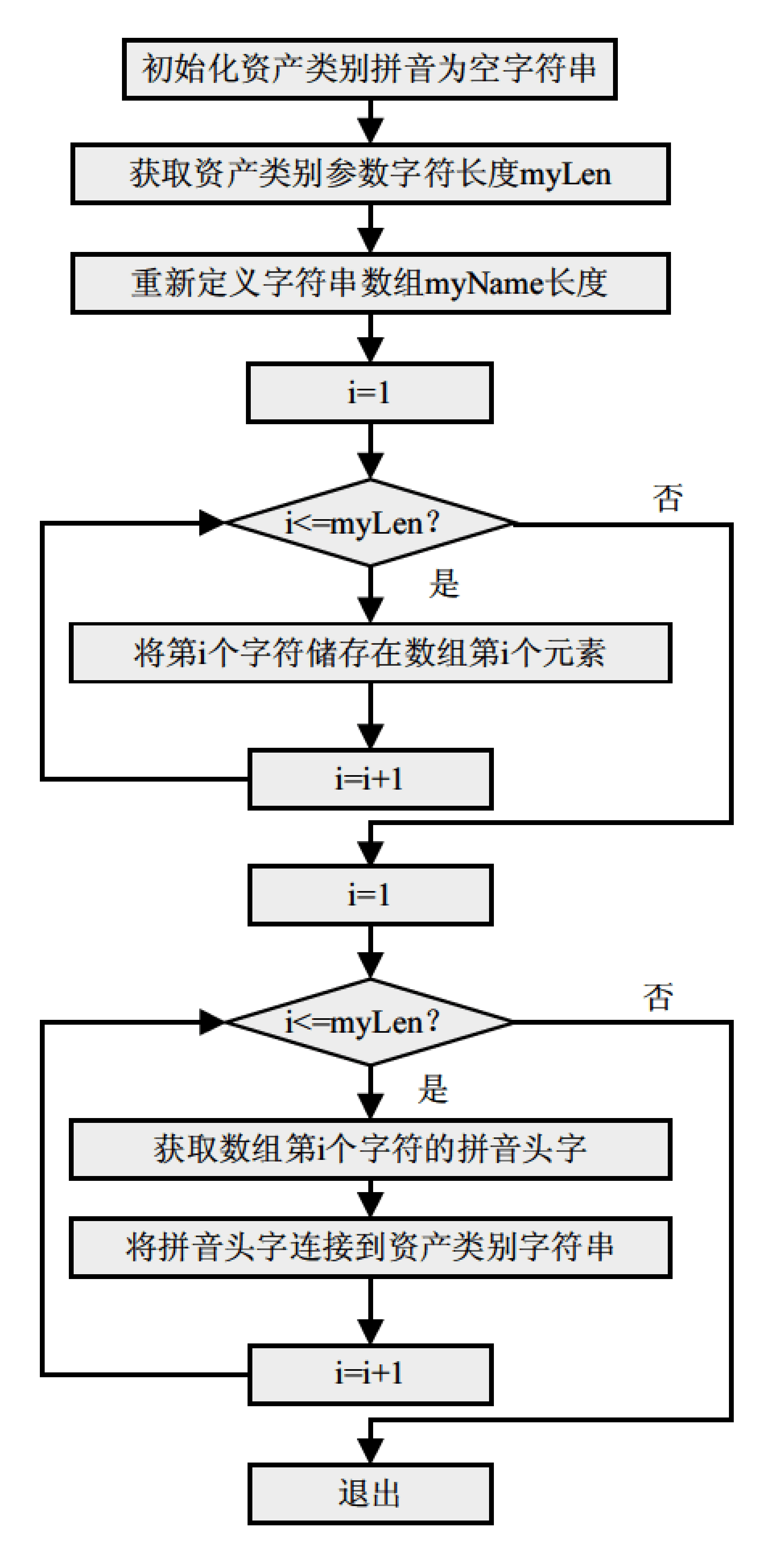


图 6-43　资产类别拼音函数流程图

该函数的详细代码解释如下：

```
Public Function 资产类别拼音(资产类别 As String) As String
    Dim myLen As Integer, i As Integer, myName() As String
    资产类别拼音 = ""                                          '初始化函数返回值
    '获取资产类别字符串的长度
    myLen = Len(资产类别)
    '资产类别的每个汉字保存到数组 myName
    ReDim myName(1 To myLen) As String
    For i = 1 To myLen                                        '循环资产类别参数字符串中每个字符
        myName(i) = Mid(资产类别, i, 1)                        '将第 i 个字符保存到数组中
    Next
    '获取客户名称的汉语拼音的第一个字母
    For i = 1 To myLen
        资产类别拼音 = 资产类别拼音 & 拼音头字母(myName(i))      '获取汉语字符的拼音头字母
                                                               并连接
    Next
End Function
```

6.12.4　拼音头字母函数代码设计

拼音头字母函数根据获取的汉字参数，返回该汉字的拼音头字母。程序通过一个多支的 If 语句逐个将汉字的 ASCII 码与字符表每个字符的首位汉字进行比较。当该汉字的 ASCII 码的值落在某个字母的首尾汉字 ASCII 码之间时，程序即会获取返回结果。如图 6-44 所示的是该函数的流程图：

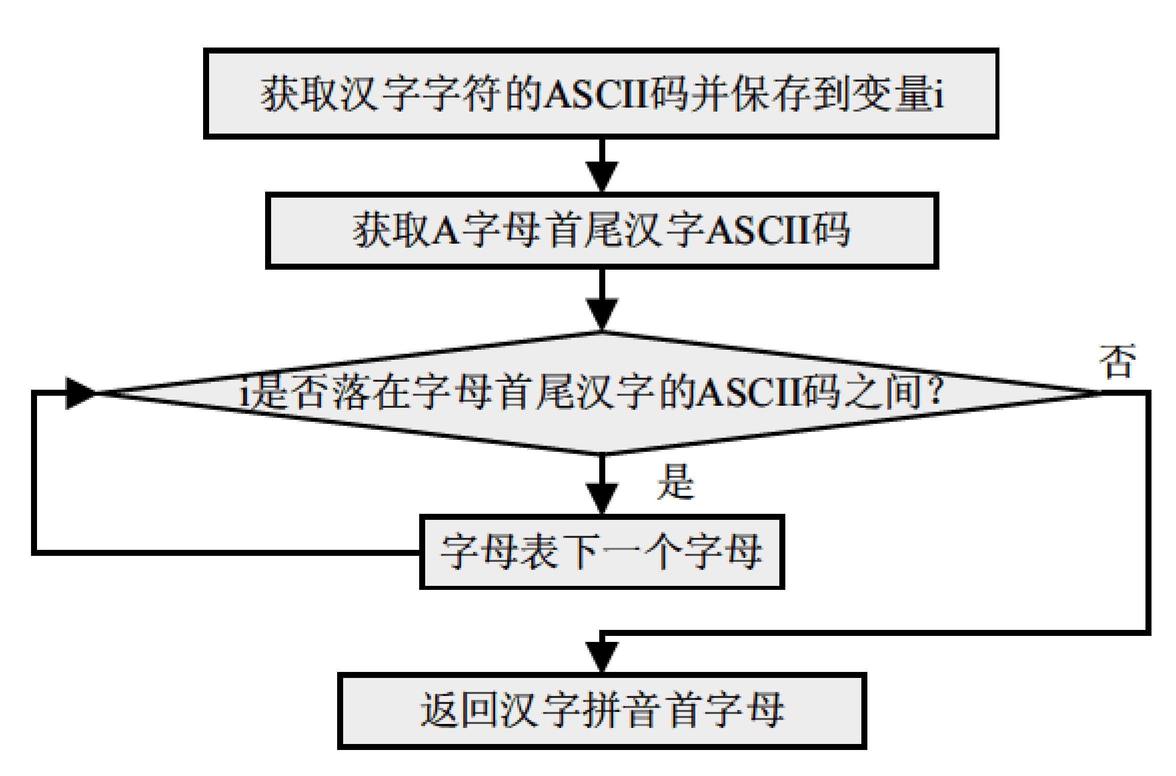


图 6-44　拼音头字母函数流程

在下面的代码中，只通过一个 If 语句来描述比较结果。

```
Public Function 拼音头字母(myChar As String) As String
    Dim i As Long
    i = Asc(myChar)                                    '获取汉字字符的 ASCII 码
    If i >= Asc("啊") And i < Asc("芭") Then            '与 A 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "A"                                '函数返回值为 A
    ElseIf i >= Asc("芭") And i < Asc("擦") Then        '与 B 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "B"                                '函数返回值为 B
    ElseIf i >= Asc("擦") And i < Asc("搭") Then        '与 C 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "C"                                '函数返回值为 C
    ElseIf i >= Asc("搭") And i < Asc("蛾") Then        '与 D 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "D"                                '函数返回值为 D
    ElseIf i >= Asc("蛾") And i < Asc("发") Then        '与 E 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "E"                                '函数返回值为 E
    ElseIf i >= Asc("发") And i < Asc("噶") Then        '与 F 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "F"                                '函数返回值为 F
    ElseIf i >= Asc("噶") And i < Asc("哈") Then        '与 G 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "G"                                '函数返回值为 G
    ElseIf i >= Asc("哈") And i < Asc("击") Then        '与 H 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "H"                                '函数返回值为 H
    ElseIf i >= Asc("击") And i < Asc("喀") Then        '与 J 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "J"                                '函数返回值为 J
    ElseIf i >= Asc("喀") And i < Asc("垃") Then        '与 K 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "K"                                '函数返回值为 K
    ElseIf i >= Asc("垃") And i < Asc("妈") Then        '与 L 字母的首尾汉字 ASCII 码比较
```

```
        拼音头字母 = "L"                                    '函数返回值为 L
    ElseIf i >= Asc("妈") And i < Asc("拿") Then          '与 M 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "M"                                    '函数返回值为 M
    ElseIf i >= Asc("拿") And i < Asc("哦") Then          '与 N 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "N"                                    '函数返回值为 N
    ElseIf i >= Asc("哦") And i < Asc("啪") Then          '与 O 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "O"                                    '函数返回值为 O
    ElseIf i >= Asc("啪") And i < Asc("欺") Then          '与 P 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "P"                                    '函数返回值为 P
    ElseIf i >= Asc("欺") And i < Asc("然") Then          '与 Q 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "Q"                                    '函数返回值为 Q
    ElseIf i >= Asc("然") And i < Asc("撒") Then          '与 R 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "R"                                    '函数返回值为 R
    ElseIf i >= Asc("撒") And i < Asc("塌") Then          '与 S 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "S"                                    '函数返回值为 S
    ElseIf i >= Asc("塌") And i < Asc("挖") Then          '与 T 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "T"                                    '函数返回值为 T
    ElseIf i >= Asc("挖") And i < Asc("昔") Then          '与 W 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "W"                                    '函数返回值为 W
    ElseIf i >= Asc("昔") And i < Asc("压") Then          '与 X 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "X"                                    '函数返回值为 X
    ElseIf i >= Asc("压") And i < Asc("匝") Then          '与 Y 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "Y"                                    '函数返回值为 Y
    ElseIf i >= Asc("匝") And i <= Asc("座") Then         '与 Z 字母的首尾汉字 ASCII 码比较
        拼音头字母 = "Z"                                    '函数返回值为 Z
    End If
End Function
```

6.12.5 获取资产编号函数代码设计

获取资产编号函数根据提供的资产类别参数确定该种类型资产的新资产编号。每一种资产的资产编号都是由两部分构成：一部分是该资产类别的拼音头字母，另外一部分是该资产的数字编号段。数字编号段的长度可以由用户在基础设置窗体中进行设置。

该函数首先获取了当前资产资产类别的拼音头字母，然后程序获取该种资产类别下固定资产的最大编号。在获取该最大编号时，首先在资产登记统计工作表中的资产编号列逐个检测。当资产编号的字母段与当前资产类型拼音头字母相同时，程序将该资产编号的数字段与最大编号进行比较，确认出新的最大编号。当获取了最大编号后，在该编号基础上加 1 即可获取该类别资产的新资产编号的数字段。

新获取的资产编号数字段只是一个数字，可能该数字的长度不够用户设置的长度。为此程序在后面的代码中为该数字段不够长度的位添加“0”字符以占位。最后程序将新资产编号的两个部分连接起来并返回给函数作为结果。该函数的流程比较复杂，以下将该函数的执行流程分为两个流程图加以解释。其中第一个流程图是整个函数的流程图，如图 6-45 所示。第二个流程图描述了整个函数过程中获取资产数字编号段的过程，如图 6-46 所示。

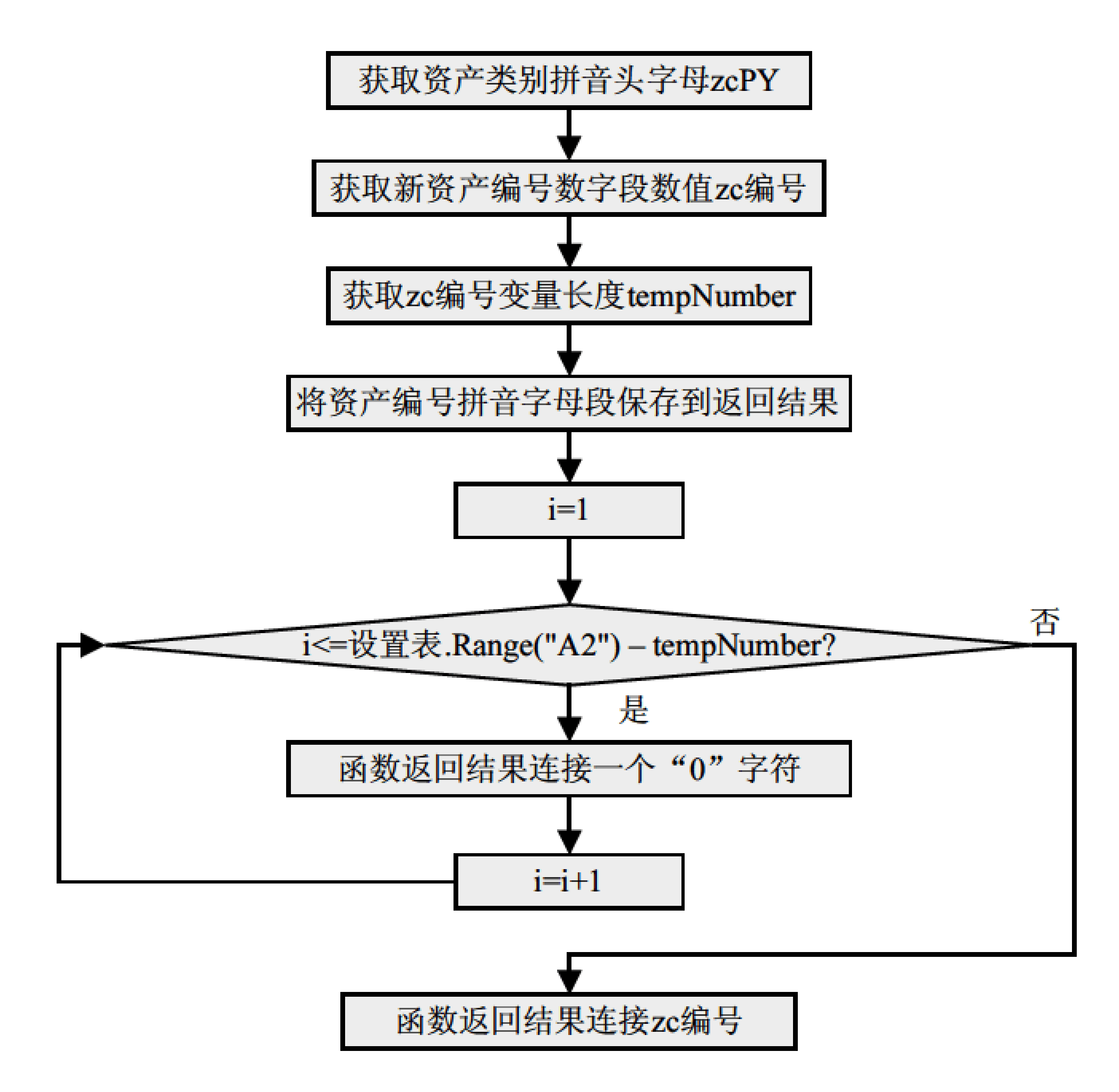


图 6-45　获取资产编号函数流程

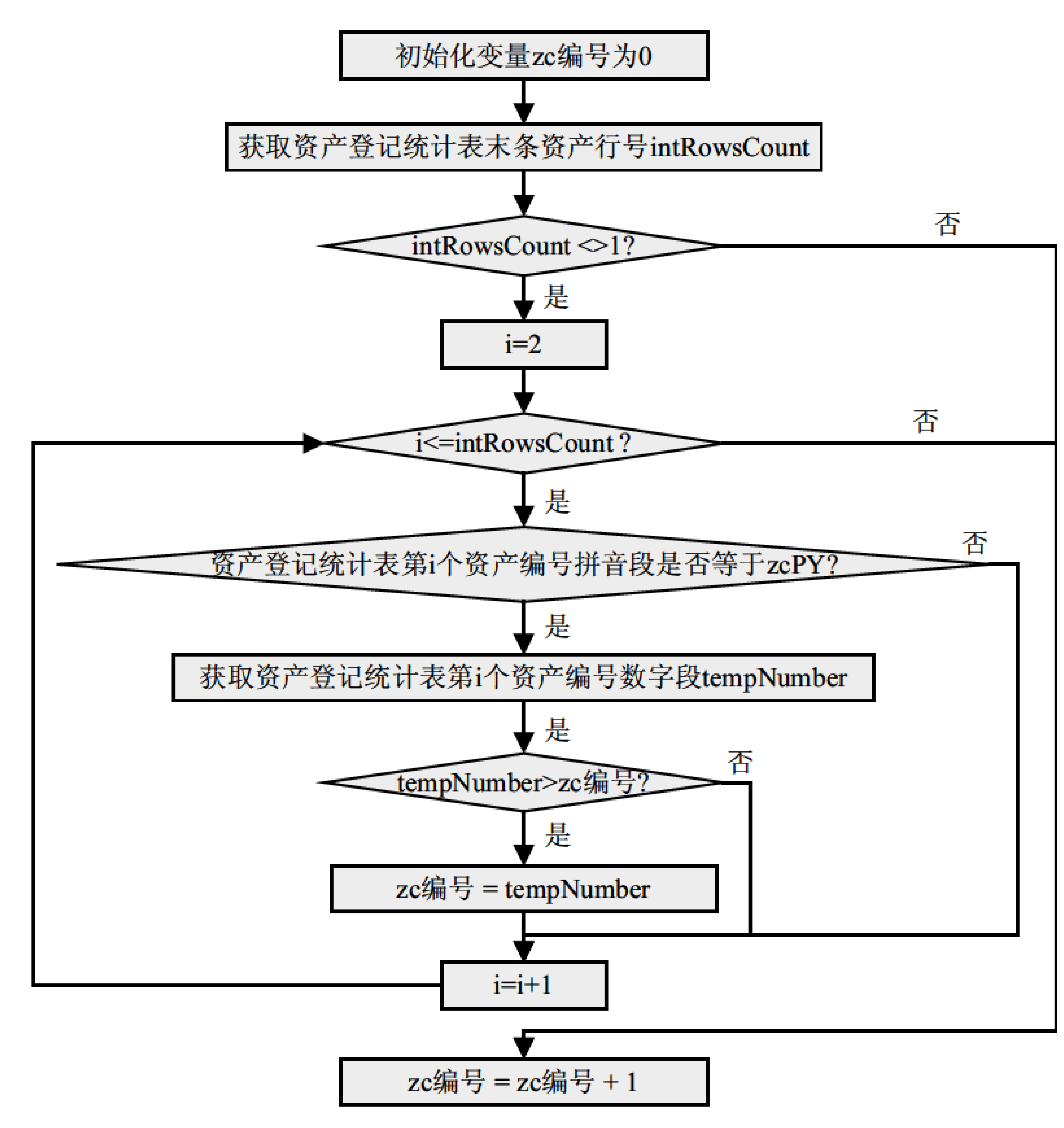


图 6-46　确定资产编号数字段流程

以下是该函数的详细代码解释：

```
Public Function 获取资产编号(资产类别 As String) As String
Dim zcPY As String, zc 编号 As Integer, intRowsCount As Integer, i As Integer
Dim tempNumber As Integer
zcPY = 资产类别拼音(资产类别)                                '获取资产编号的拼音字母段
zc 编号 = 0                                                 '初始化资产编号数字段数值
'获取资产登记统计表末条记录行号
intRowsCount = 资产登记统计.Range("C" & Rows.Count).End(xlUp).Row
If intRowsCount <> 1 Then                                   '检测资产登记统计表记录是否超过一个
    For i = 2 To intRowsCount                               '循环资产登记统计表所有记录
        '检测第 i 行资产是否属于当前资产类别
        If Left(资产登记统计.Range("C" & i), Len(资产类别)) = zcPY Then
            '保存第 i 行资产编号的数字段数值
            tempNumber = CInt(Right(资产登记统计.Range("C" & i), Len(资产登记统计.Range("C" &
i)) -Len(资产类别)))
            '确认最大资产编号数值
            If tempNumber > zc 编号 Then zc 编号 = tempNumber
        End If
    Next
End If
zc 编号 = zc 编号 + 1                                        '生成新资产编号的数字段数值
tempNumber = Len(CStr(zc 编号))                              '获取数字段数值的位数
获取资产编号 = 获取资产编号 & zcPY                            '函数返回结果连接拼音字符串
For i = 1 To 设置表.Range("A2") - tempNumber                 '循环产生固定位数的资产编号数字段
    获取资产编号 = 获取资产编号 & "0"                          '函数返回结果连接一个“0”
Next
获取资产编号 = 获取资产编号 & zc 编号                          '函数返回结果连接数字段数值
End Function
```

6.12.6 计提折旧过程代码设计

计提折旧过程用于对某项固定资产计提折旧，该过程是在用户设置计提折旧日期之后被运行的。计提折旧时，必须对所有已登记固定资产计提折旧。

为了实现计提折旧功能，程序从资产登记统计工作表中获取需要计提折旧的资产名。然后程序由该资产名获取对应该资产的工作表对象，随后在该资产折旧明细表中检测该资产是否需要计提折旧。当确定需要计提折旧后，程序在该折旧明细表中添加一行新折旧信息。在填写固定资产折旧额时，还需要确定剩余折旧额小于月折旧额，否则直接填入月折旧额时会造成计提的实际总折旧额超过预定总折旧额，随之该固定自残净值出现负值。如图 6-47 所示的是该过程的流程图。

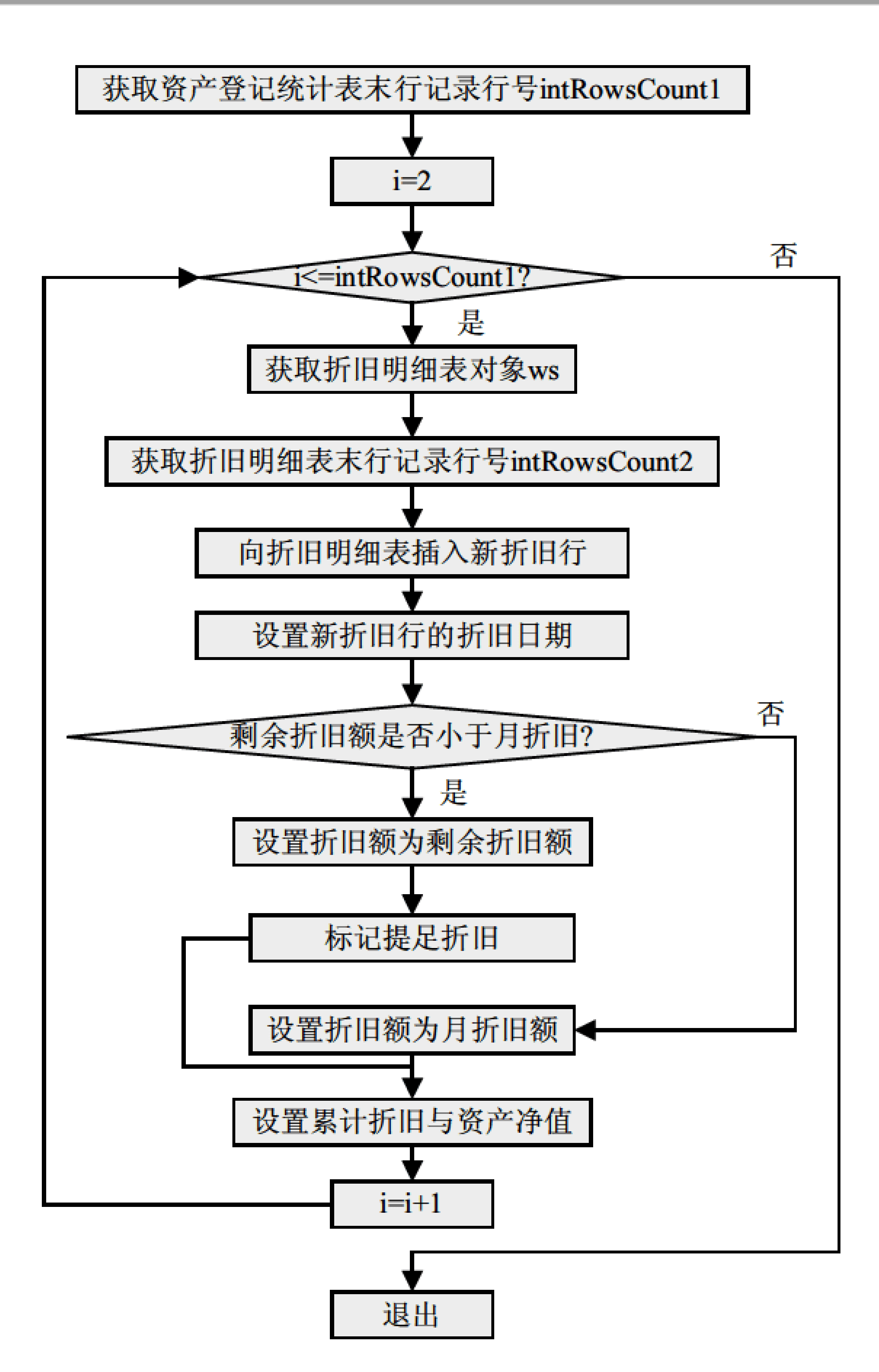


图 6-47　计提折旧过程流程图

以下是该过程的详细代码解释：

```
Sub 计提折旧()
Dim ws As Worksheet, intRowsCount1 As Integer, i As Integer, intRowsCount2 As Integer
'获取资产登记统计表末行记录行号
intRowsCount1 = 资产登记统计.Range("B" & Rows.Count).End(xlUp).Row
For i = 2 To intRowsCount1                              '循环资产登记统计表中所有资产
    '获取资产折旧明细表对象
    Set ws = Worksheets("MX-" & 资产登记统计.Range("C" & i) & "-" & 资产登记统计.Range("D" & i))
    With ws
        '获取资产折旧明细表末行记录行号
        intRowsCount2 = .Range("A" & Rows.Count).End(xlUp).Row
        If 是否计提(ws, 设置表.Range("D2")) Then        '检测固定资产是否需要计提折旧
            '为固定资产折旧明细表插入新折旧行
            ws.Rows(intRowsCount2 + 1).Insert shift:=xlDown, CopyOrigin:=xlFormatFromLeftOrAbove
            '设置新折旧行折旧日期
            .Range("A" & (intRowsCount2 + 1)) = Format(设置表.Range("D2"), "YYYY-M-D")
            '检测剩余折旧额是否小于月折旧额
            If .Range("B6") -.Range("K" & intRowsCount2) -.Range("L4") < .Range("L6") Then
                '设置折旧额为剩余折旧额
```

```
                .Range("J" & (intRowsCount2 + 1))=.Range("B6")-.Range("K" & intRowsCount2)-
.Range("L4")
                ws.Range("K8") = "提足折旧"              '在折旧明细表中设置已提足折旧
                资产登记统计.Range("S" & i) = "提足折旧"   '在资产登记统计表中设置已提足折旧
            Else
                .Range("J" & (intRowsCount2 + 1)) = .Range("L6")       '设置折旧额为月折旧额
            End If
            '设置累计折旧与资产净值
            .Range("K" & (intRowsCount2 + 1)) = .Range("K" & intRowsCount2) + .Range("L6")
            .Range("L" & (intRowsCount2 + 1)) = .Range("B6") -.Range("K" & (intRowsCount2 + 1))
        End If
    End With
Next
End Sub
```

6.12.7　是否计提函数代码设计

是否计提函数用于确认某项固定资产是否需要计提某个月的折旧。是否对某个固定资产在某个日期需要计提折旧，影响的因素比较多。这些因素包括以下几个方面：

- ❑ 固定资产已经计提该月折旧：当用户为某固定资产建立了某个月的折旧后，该固定资产的明细折旧表中会登记该月的折旧信息。这种情况下，程序需要检测是否有折旧记录存在。
- ❑ 折旧日期必须在自残使用日期后：固定资产折旧的日期一般应该设置在使用日期以后。此时需要检测该折旧日期是否在资产使用日期之前。
- ❑ 资产折旧已经计提完毕：当固定资产的折旧已经计提完毕，此时再继续计提折旧没有意义。

函数对于以上 3 种情况，分别使用了 3 个 If 语句。对于第一种情况，程序需要在固定资产明细折旧表中逐行检测，因而使用了一个 For 循环。该函数的整体结构比较简单，如图 6-48 所示的是该函数的流程图。

图 6-48　是否计提函数流程图

以下是该函数的详细代码解释：

```
Function 是否计提(ws As Worksheet, 日期 As Date) As Boolean
Dim intRowsCount As Integer, i As Integer
```

```
intRowsCount = ws.Range("A" & Rows.Count).End(xlUp).Row      '获取工作表末行记录行号
是否计提 = True                                                '设置函数返回为真
For i = 11 To intRowsCount                                    '循环检测计提折旧日期列所有记录
    '检测当前日期是否已经计提折旧
    If Format(ws.Range("A" & i), "yyyy-m") = Format(日期, "YYYY-M") Then
        是否计提 = False                                        '设置不计提折旧
        Exit Function                                         '退出函数
    End If
Next
'确定计提日期不在资产使用日期前
If Year(ws.Range("F6")) > Year(日期) _
    Or (Year(ws.Range("F6")) = Year(日期) And Month(ws.Range("F6")) >= Month(日期)) Then
    是否计提 = False                                            '设置不计提折旧
    Exit Function                                             '退出函数
End If
If ws.Range("K8") = "提足折旧" Then                             '检测是否计提完毕折旧
    是否计提 = False                                            '设置不计提折旧
    Exit Function                                             '退出函数
End If
End Function
```

6.13　系 统 测 试

本小节对系统中几个比较重要的功能进行测试，这些功能包括固定资产登记、查看固定资产信息、计提折旧和固定资产折旧与现值统计。以下将分 4 个小节，以一个固定资产管理范例讲述这些功能的操作过程。实例中建立的固定资产是一部传真机。

6.13.1　固定资产登记

（1）在首页单击【固定资产登记】按钮，系统自动跳转到固定资产登记表中并清除所有数据，如图 6-49 所示。

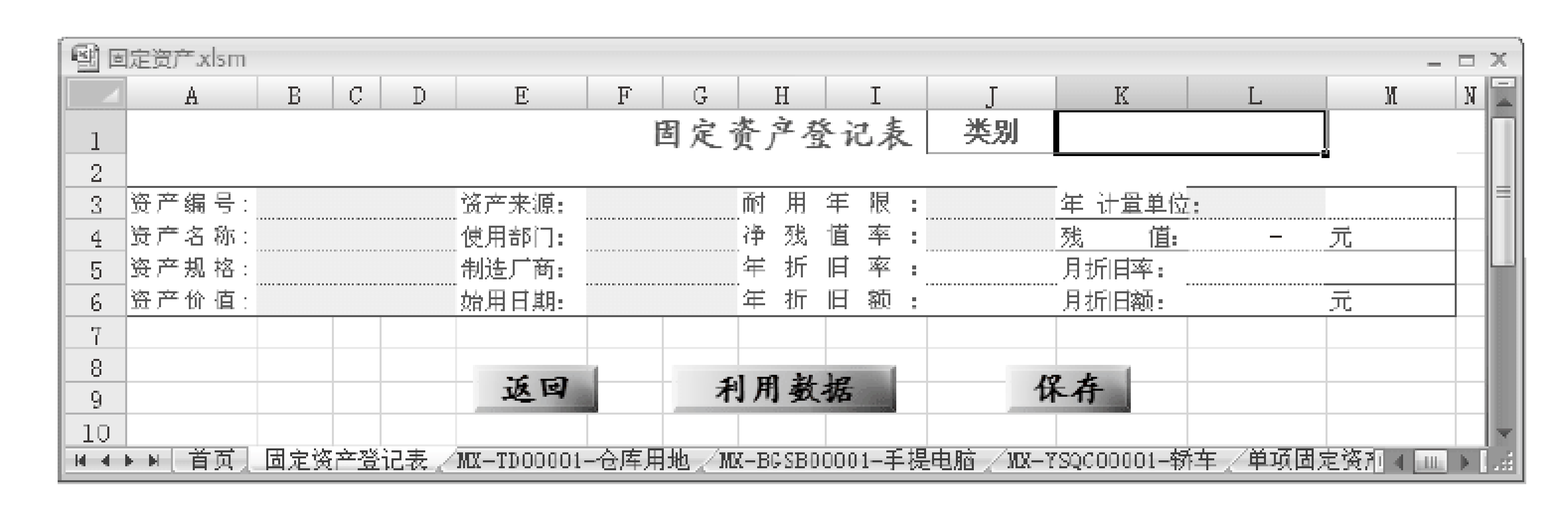


图 6-49　固定资产登记表

（2）双击固定资产登记表类别栏 K1 单元格，打开【资产类别】对话框，如图 6-50 所示。选择办公设备，此时系统将自动获取资产编号 BGSB00002。在资产名称 B4 单元格中输入“联想台式机”，资产价值 B6 单元格中输入 7000。双击使用部门 F4 单元格，打开【使用部门】

对话框，如图 6-51 所示。选择财务部，在始用日期 F6 单元格中输入“2007-12-10”，耐用年限 J3 单元格中输入 5。输入完成后的结果如图 6-52 所示。最后单击【保存】按钮保存该固定资产信息。





图 6-50 【资产类别】对话框　　图 6-51 【使用部门】对话框

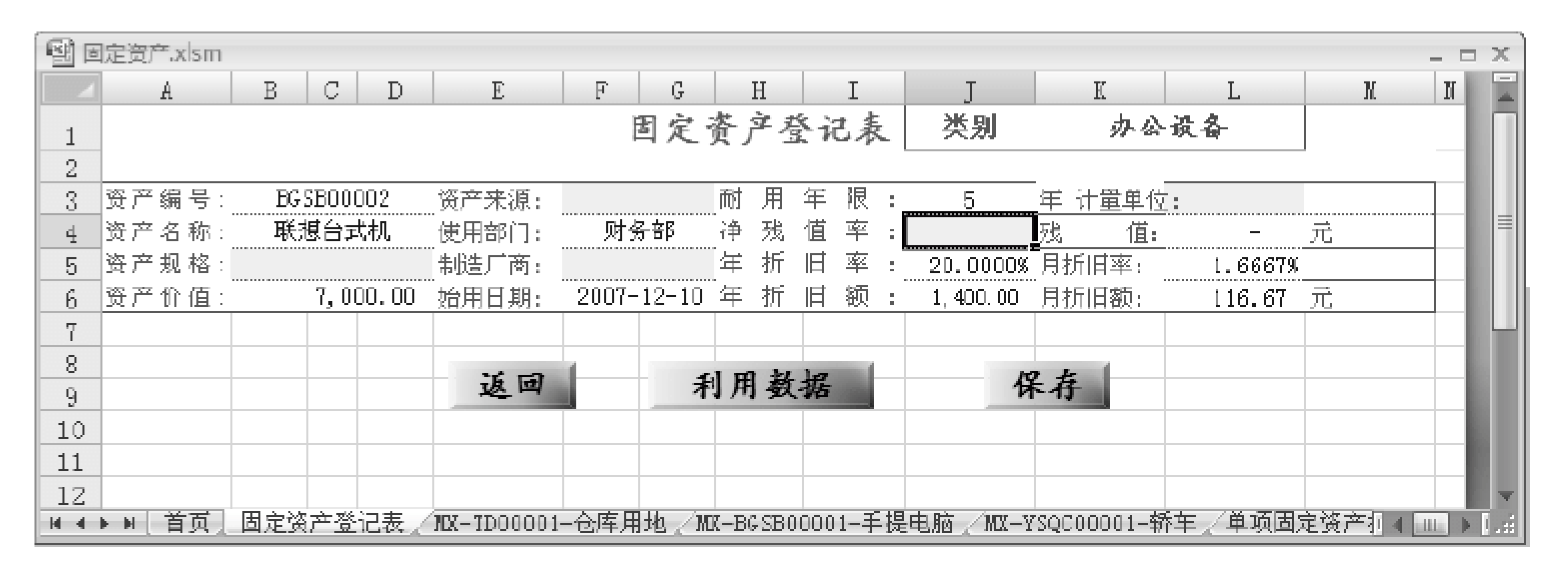


图 6-52 登记固定资产信息

6.13.2 查看资产信息

首先返回首页，然后在首页中单击【查看单项固定资产】按钮，打开【选择明细表】对话框，如图 6-53 所示。选择最后一个项目，此时系统自动跳转到资产明细信息表，如图 6-54 所示。该明细表中包含了固定资产的分类账。





图 6-53 【选择明细表】对话框　　图 6-54 固定资产明细信息表

6.13.3 计提折旧

（1）返回系统首页，在首页中单击【计提折旧】按钮，在弹出的【设置计提日期】对话



图 6-55　【设置计提日期】对话框

框中设置计提日期，将年份设置为 2007 年，月份设置为 12 月，如图 6-55 所示，单击【确定】按钮即可。

（2）系统计提折旧完成后，会自动弹出一个提示窗口，如图 6-56 所示。单击【确定】按钮，可以继续计提折旧，单击【取消】按钮可以退出折旧操作。这里还需要计提 2008 年 1 月和 2 月的折旧额，因而单击【确定】按钮，同第一步操作，分别计提 1 月和 2 月的折旧即可。所有折旧完成后，可以查看一下刚登记的固定资产的折旧情况，按照 6.13.2 节步骤打开刚登记固定资产的明细表即可，如图 6-57 所示。



图 6-56　提示窗口

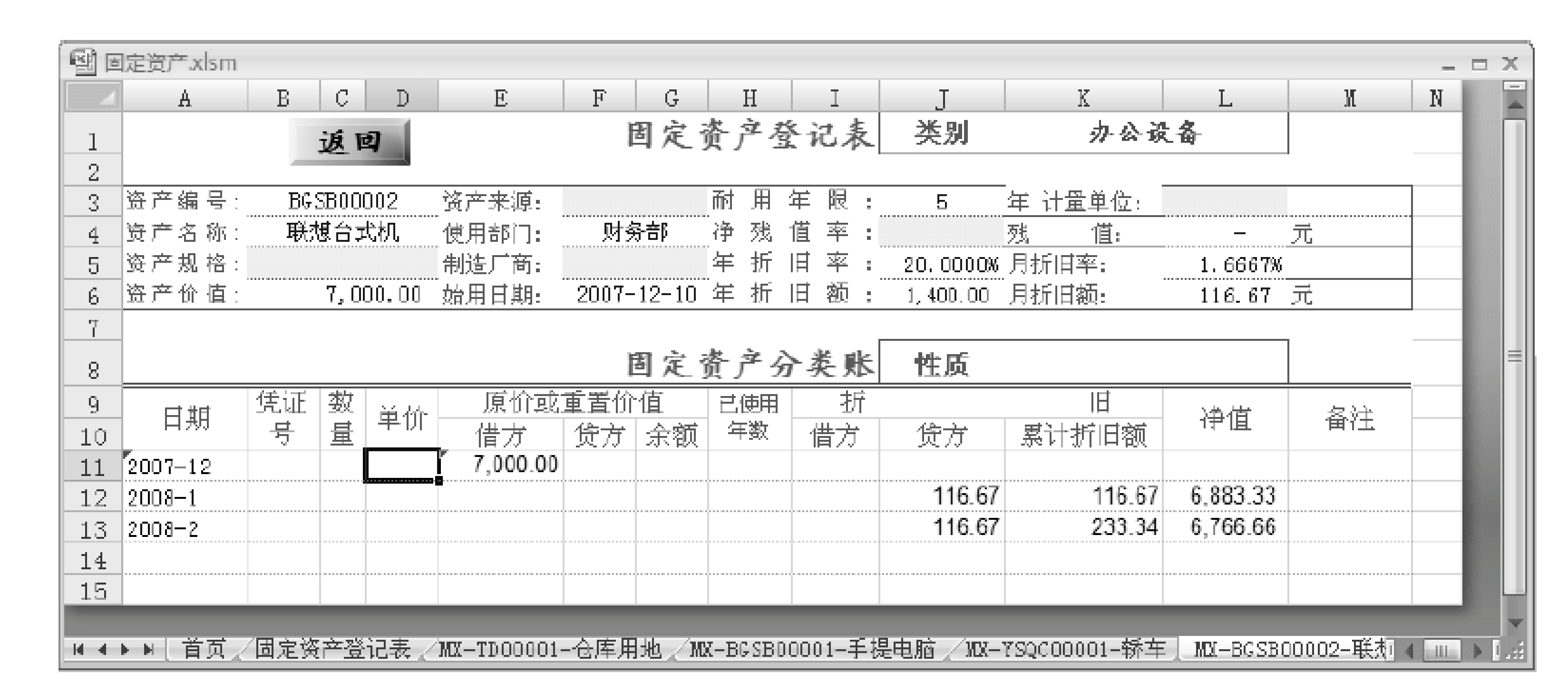


图 6-57　资产折旧明细表

6.13.4　固定资产折旧与现值

固定资产折旧与现值操作十分简单，只需返回到首页并单击【资产折旧与现值】按钮即可，程序将自动完成所有分析统计显示工作。完成以上操作后，固定资产的折旧与现值统计结果如图 6-58 所示。

固定资产折旧与现值统计

资产编号	资产名称	原价	月折	月折旧额	折旧总额	净值
TD00001	仓库用地	400000	0.00	1,055.56	23,222.32	376,777.68
BGSB00001	手提电脑	10000	0.02	161.67	2,101.71	7,898.29
YSQC00001	轿车	100000	0.00	395.83	3,958.30	96,041.70
BGSB00002	联想台式机	7,000.00	1.67%	116.67	233.34	6766.66
合　计		517,000.00		1,729.73	29,515.67	487,484.33

返回

MX-BGSB00002-联想台式机　单项固定资产折旧明细模板　固定资产登记统计　固定资产折旧

图 6-58　固定资产折旧与现值统计表

第 7 章　进销存管理系统

对于一般的商贸公司，进货、销售与存货的日常处理非常频繁，这些工作也是大多数企业经营的管理核心，数据的变动性非常大。购进什么商品、购入多少，如何减少库存积压等问题都会被这些数据所影响。如果通过人来记忆，很难保证这些数据的正确性，从而很可能造成断货、商品积压问题。使用进销存管理系统可以及时了解进销存状况，快速做出反应。

7.1　系 统 概 述

本系统是一个小型的进销存管理系统，通过该系统可以轻松完成各项进销存操作以及各项相关查询。通过该系统获得的日常进销存数据，可以快速获得企业进销存状况。

7.1.1　设计思路

系统共包含了系统管理、基本资料管理、进货管理、销售管理、库存管理、资料查询与导出 6 个功能模块，分别对应了该系统的 6 个功能菜单。在这些功能菜单中分别又包含了各自的子功能菜单。系统的功能结构图如图 7-1 所示。

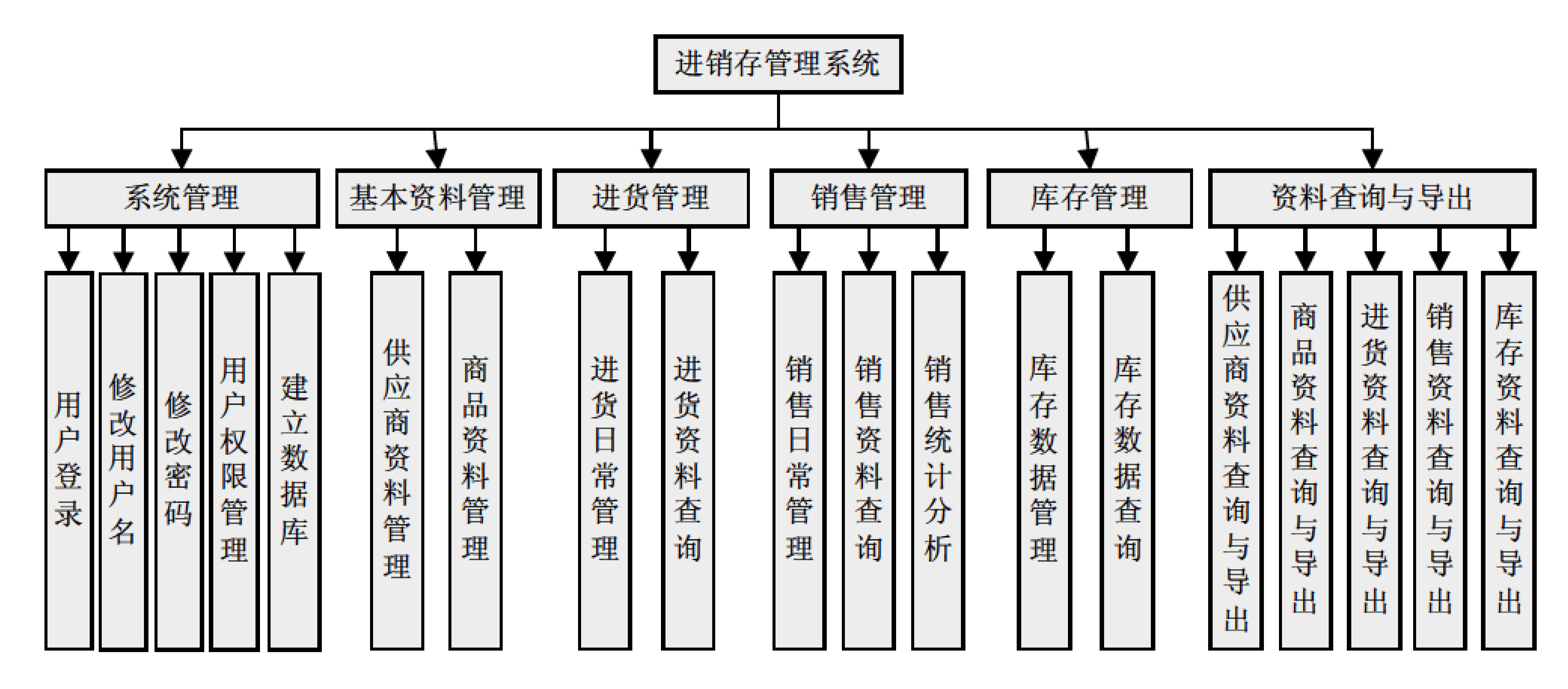


图 7-1　进销存管理系统功能模块结构图

各模块的详细功能介绍如下：

- 系统管理模块：该模块用于用户登录、用户名修改、用户密码修改、用户权限设置以及创建数据库。
- 基本资料管理模块：该模块用于完成供货商基本资料和商品基本资料的添加、修改

和删除等基本操作。

- 进货管理模块：该模块用于完成日常进货相关操作，例如添加、修改和删除等进货基本操作。该模块也可以完成查询与导出操作。
- 销售管理模块：该模块用于完成日常销售相关操作，例如添加、修改和删除等销售基本操作。该模块也可以完成查询与导出操作。
- 库存管理模块：该模块用于完成日常库存相关操作，例如查询库存与汇总操作。
- 资料查询与导出模块：在该模块中可以集中完成对供货商资料、商品资料、进货、销售、库存的查询和导出操作。

7.1.2　知识点：自定义菜单

在 Excel 2007 中通过代码添加自定义菜单的方法很方便。首先使用 MenuBars 的 Add 方法添加菜单栏，然后使用该菜单栏的 Menus 属性的 Add 方法添加一级菜单。添加二级菜单时，应该使用一级菜单的 MenuItems 属性的 Add 方法。下面是一个简单的实例：

```
Sub Auto_Open()
MenuBars.Add "自定义菜单"
'为自定义菜单栏添加菜单
MenuBars("自定义菜单").Menus.Add "一级菜单"
'为菜单添加子菜单，并指定宏
With MenuBars("自定义菜单").Menus("一级菜单").MenuItems
        .Add "二级菜单-1", "二级菜单-1 执行过程"
        .Add "-"
        .Add "二级菜单-2", "二级菜单-2 执行过程"
End With
End Sub
```

通常对于自定义的菜单栏，在退出工作簿时，应该清除该自定义工具栏。不完成这些操作，很容易造成菜单栏混乱。删除的方法如下：

```
Private Sub Workbook_BeforeClose(Cancel As Boolean)
    MenuBars("自定义菜单").Delete
End Sub
```

7.2　Access 数据库设计

本系统以 Access 数据库作为后台数据库，在本系统的 Excel 表中不保存相关数据，但是用户的密码设置保存在工作表中。其中的数据都通过 ADO 数据库对象操作。

7.2.1　数据表设计

在建立该 Access 数据前，首先需要完成的工作就是设计各个表包含的字段。该数据库共

包含了 5 个数据表，分别是供货商信息、商品信息、进货信息、销售信息和库存信息。各个表的设计状况如表 7-1~表 7-5 所示。

表 7-1　供货商信息表字段设计

字段名称	数据类型	字段长度	是否允许为空
供货商编码	文本	10	否
供货商名称	文本	40	否
通讯地址	文本	30	否
邮政编码	文本	6	否
联系电话	文本	14	否
传真号码	文本	14	否
联系人	文本	10	否
联系人电话	文本	14	否
联系人 Email	文本	50	否
备注	文本	50	是

表 7-2　商品信息表字段设计

字段名称	数据类型	字段长度	是否允许为空
商品编码	文本	10	否
商品名称	文本	20	否
商品规格	文本	10	否
计量单位	文本	10	否
最高库存	单精度		否
最低库存	长整型		否
备注	文本	50	是

表 7-3　进货信息表字段设计

字段名称	数据类型	字段长度	是否允许为空
进货编码	文本	10	否
供货商编码	文本	10	否
商品编码	文本	10	否
商品名称	文本	20	否
商品规格	文本	10	否
计量单位	文本	10	否
进货数量	单精度		否
进货单价	单精度		否
进货日期	日期		否
备注	文本	50	是

表 7-4　销售信息表字段设计

字 段 名 称	数 据 类 型	字 段 长 度	是否允许为空
销售编码	文本	10	否
商品编码	文本	10	否
商品名称	文本	20	否
商品规格	文本	10	否
计量单位	文本	10	否
销售数量	单精度		否
销售单价	单精度		否
销售日期	日期		否
备注	文本	50	是

表 7-5　库存信息表字段设计

字 段 名 称	数 据 类 型	字 段 长 度	是否允许为空
商品编码	文本	10	否
商品名称	文本	20	否
商品规格	文本	10	否
计量单位	文本	10	否
库存数量	单精度		否
库存单价	单精度		否
库存金额	单精度		否

7.2.2　建立数据库代码

Access 数据库文件中的各个表的结构，在该系统中是通过代码来生成的，这个工作不需要手动完成。在程序所提供的菜单中选择【系统管理】|【建立数据库】命令，系统会自动生成所有数据表（表中没有任何记录）。当选择该菜单后，系统执行了创建数据库过程。该过程的流程图如图 7-2 所示。

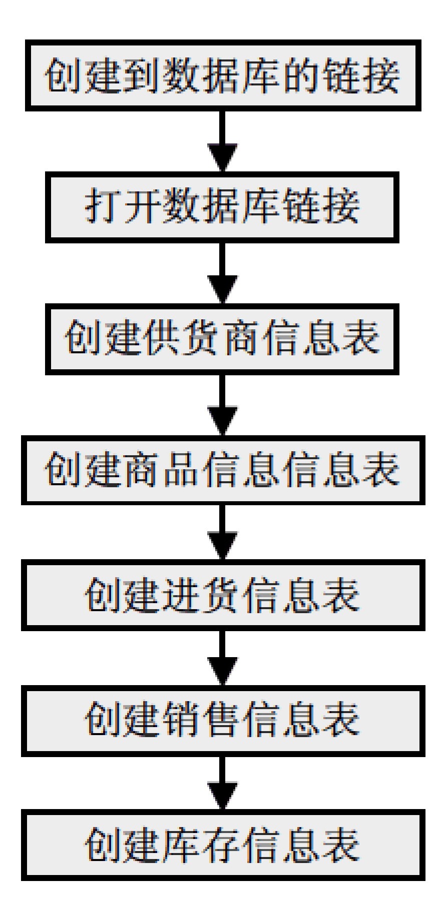
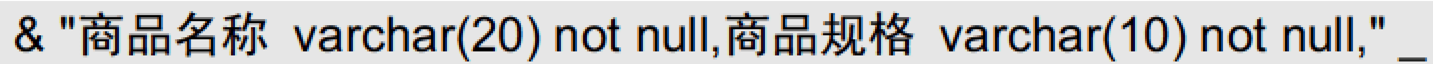


图 7-2　创建数据库流程图

该过程的详细代码解释如下：

```
Public Sub 创建数据库()
    Dim cnn As New ADODB.Connection
    Dim adoxCat As New ADOX.Catalog
    Dim cnnStr As String, SQL As String
    '创建与 Access 数据库链接的 Connection 对象
    cnnStr = ThisWorkbook.Path & "\进销存数据库.mdb"
    '判断数据库是否存在
    If Dir(cnnStr) <> "" Then                                    '检测数据库文件是否存在
        MsgBox "数据库已经存在！", vbInformation, "检查数据库"
        Set cnn = Nothing
        Set adoxCat = Nothing
        Exit Sub
    End If
    '建立 Access 数据库
    cnnStr = "Provider=microsoft.jet.oledb.4.0;" _
        & "Data Source=" & ThisWorkbook.Path & "\进销存数据库.mdb;" _
        & "Jet Oledb:database password=123456;"                  '设置数据库创建字符串
    adoxCat.Create cnnStr                                        '链接数据库
    Set mycat = Nothing
    '创建数据库链接对象
    cnn.ConnectionString = cnnStr
    cnn.Open
    '创建数据表“供货商信息”
    SQL = "create table 供货商信息" _
        & "(供货商编码 varchar(10) not null,供货商名称 varchar(40) not null," _
        & "通讯地址 varchar(30) not null,邮政编码 varchar(6) not null," _
        & "联系电话 varchar(14) not null,传真号码 varchar(14) not null," _
        & "联系人 varchar(10) not null,联系人电话 varchar(14) not null," _
        & "联系人 Email varchar(50) not null,备注 varchar(50))"      '设置创建供货商信息表字符串
    cnn.Execute SQL                                              '创建供货商信息表
    '创建数据表“商品信息”
    SQL = "create table 商品信息" _
        & "(商品编码 varchar(10) not null,商品名称 varchar(20) not null," _
        & "商品规格 varchar(10) not null,计量单位 varchar(10) not null," _
        & "最高库存 single not null,最低库存 int not null,备注 varchar(50))"   '设置创建商品信息表
                                                                     字符串
    cnn.Execute SQL                                              '创建商品信息表
    '创建数据表“进货信息”
    SQL = "create table 进货信息" _
        & "(进货编码 varchar(10) not null,供货商编码 varchar(10) not null," _
        & "商品编码 varchar(10) not null,商品名称 varchar(20) not null," _
        & "商品规格 varchar(10) not null,计量单位 varchar(10) not null," _
        & "进货数量 single not null,进货单价 real not null," _
        & "进货日期 datetime not null,备注 varchar(50))"             '设置创建进货信息表字符串
    cnn.Execute SQL                                              '创建进货信息表
    '创建数据表“销售信息”
    SQL = "create table 销售信息" _
        & "(销售编码 varchar(10) not null,商品编码 varchar(10) not null," _
```



```
        & "商品名称  varchar(20) not null,商品规格  varchar(10) not null," _
        & "计量单位  varchar(10) not null,销售数量  single not null," _
        & "销售单价  real not null,销售日期  datetime not null,备注  varchar(50))"
                                                        '设置创建销售信息表字符串
    cnn.Execute SQL                                     '创建销售信息表
    '创建数据表“库存信息”
    SQL = "create table 库存信息" _
        & "(商品编码  varchar(10) not null,商品名称  varchar(20) not null," _
        & "商品规格  varchar(10) not null,计量单位  varchar(10) not null," _
        & "库存数量  single not null,库存单价  real not null,库存金额  real not null)"
                                                        '设置创建库存信息表字符串
    cnn.Execute SQL                                     '创建库存信息表
    MsgBox "数据库创建成功!", vbInformation, "创建数据库"
    cnn.Close
    Set cnn = Nothing
End Sub
```

7.3　系统自定义菜单

由于本系统的功能模块分类比较多，而且每个功能分类包含的子功能块也比较多，所以系统采用了自定义菜单形式完成整个系统的导航操作。在 Excel 2007 中，生成系统自定义菜单在加载项中，效果图如图 7-3 所示。

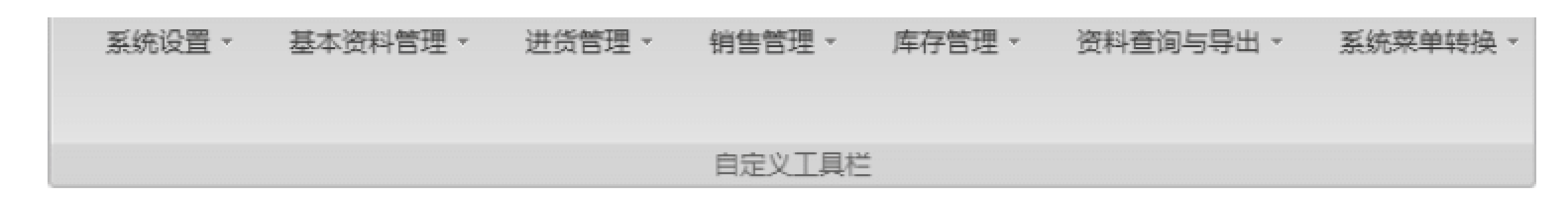


图 7-3　系统自定义菜单

7.3.1　子菜单设计

主菜单下分别对应了各自的子菜单，子菜单的内容与该系统的结构示意图相似。如图 7-4~图 7-9 所示的是各个子菜单的图例。



图 7-4　系统设置子菜单

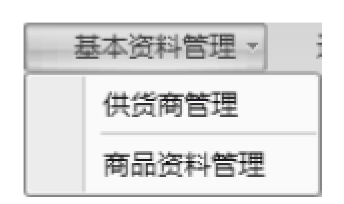


图 7-5　基本资料管理子菜单

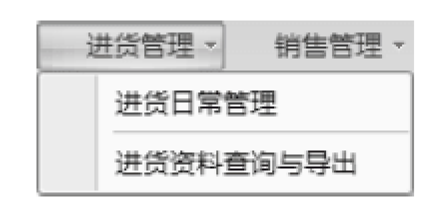


图 7-6　进货管理子菜单

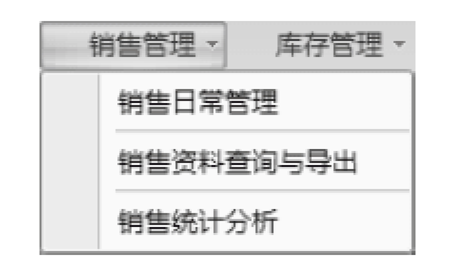


图 7-7　销售管理子菜单

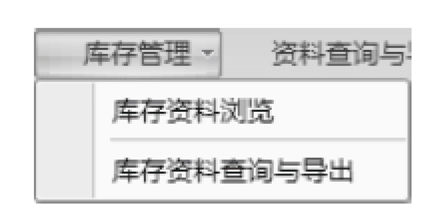


图 7-8　库存管理子菜单

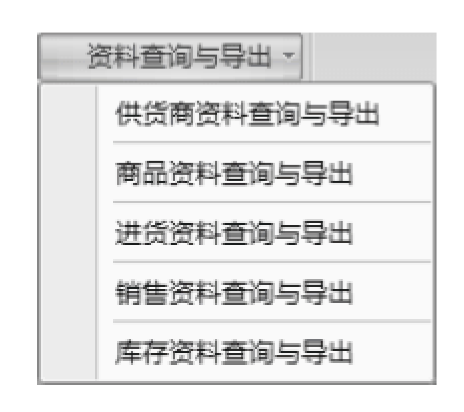


图 7-9　资料查询与导出子菜单

7.3.2　自定义菜单代码设计

以上的自定义菜单的显示结构都是通过代码实现的。代码包括创建自定义菜单以及其他所有子菜单被单击时执行的过程。如图 7-10 所示的是创建自定义菜单过程的流程图。

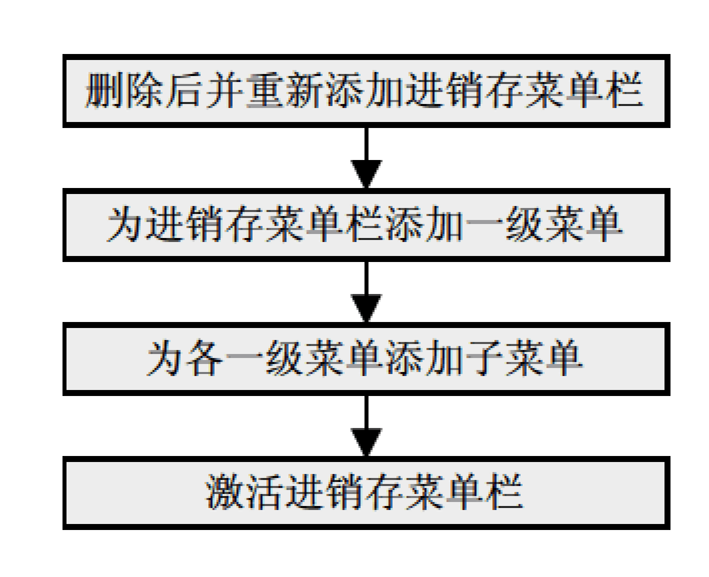


图 7-10　创建自定义菜单流程图

以下是该过程的详细代码解释：

```
Public Sub 创建自定义菜单()
    On Error Resume Next
    '清除进销存菜单栏
    MenuBars("进销存").Delete
    '建立一个自定义菜单栏“进销存”
    MenuBars.Add "进销存"
    '为“进销存”自定义菜单栏添加菜单
    With MenuBars("进销存").Menus
        .Add "系统设置"
        .Add "基本资料管理"
        .Add "进货管理"
        .Add "销售管理"
        .Add "库存管理"
        .Add "资料查询与导出"
    End With
    '为各个菜单添加子菜单
    '为“系统设置”菜单添加子菜单，并指定宏
    With MenuBars("进销存").Menus("系统设置").MenuItems
        .Add "用户登录", "用户登录窗口"        '添加菜单“用户登录”，执行的宏为“用户登录窗口”
        .Add "-"                               '添加菜单分割线
        .Add "修改用户名", "修改用户名窗口"
        .Add "-"
        .Add "修改密码", "修改密码窗口"
        .Add "-"
        .Add "用户权限管理", "用户权限窗口"
        .Add "-"
        .Add "创建数据库", "创建数据库"
    End With
    '为“基本资料管理”菜单添加子菜单，并指定宏
    With MenuBars("进销存").Menus("基本资料管理").MenuItems
        .Add "供货商管理", "供货商管理窗口"
```

```
            .Add "-"
            .Add "商品资料管理", "商品资料管理窗口"
        End With
        '为“进货管理”菜单添加子菜单，并指定宏
        With MenuBars("进销存").Menus("进货管理").MenuItems
            .Add "进货日常管理", "进货管理窗口"
            .Add "-"
            .Add "进货资料查询与导出", "资料查询与导出窗口"
        End With
        '为“销售管理”菜单添加子菜单，并指定宏
        With MenuBars("进销存").Menus("销售管理").MenuItems
            .Add "销售日常管理", "销售管理窗口"
            .Add "-"
            .Add "销售资料查询与导出", "资料查询与导出窗口"
            .Add "-"
            .Add "销售统计分析", "销售统计分析窗口"
        End With
        '为“库存管理”菜单添加子菜单，并指定宏
        With MenuBars("进销存").Menus("库存管理").MenuItems
            .Add "库存资料浏览", "库存资料浏览窗口"
            .Add "-"
            .Add "库存资料查询与导出", "资料查询与导出窗口"
        End With
        '为“资料查询与导出”菜单添加子菜单，并指定宏
        With MenuBars("进销存").Menus("资料查询与导出").MenuItems
            .Add "供货商资料查询与导出", "资料查询与导出窗口"
            .Add "-"
            .Add "商品资料查询与导出", "资料查询与导出窗口"
            .Add "-"
            .Add "进货资料查询与导出", "资料查询与导出窗口"
            .Add "-"
            .Add "销售资料查询与导出", "资料查询与导出窗口"
            .Add "-"
            .Add "库存资料查询与导出", "资料查询与导出窗口"
        End With
        '激活“进销存”菜单栏
        MenuBars("进销存").Activate
End Sub
```

对于每个子菜单而言，都有其将执行的宏过程，这些过程的代码都比较简单。以下是这些过程的详细代码解释：

```
Public Sub 用户登录窗口()
    登录.Show
End Sub

Public Sub 修改用户名窗口()
'在打开修改用户名窗口前，检测当前是否有用户登录
If Len(登录用户.用户名) Then                    '检测是否有用户登录
    修改用户名.Show                            '显示修改用户名窗口
Else
```

```
    MsgBox "您尚未登录系统，请首先登录系统！", vbOKOnly + vbInformation        '提示未登录
End If
End Sub

Public Sub 修改密码窗口()
'在打开修改密码窗口前，检测当前是否有用户登录
If Len(登录用户.用户名) Then                                   '检测是否有用户登录
    修改密码.Show                                              '显示修改密码窗口
Else
    MsgBox "您尚未登录系统，请首先登录系统！", vbOKOnly + vbInformation        '提示未登录
End If
End Sub

Public Sub 用户权限窗口()
'在打开用户权限管理窗口前，检测当前是否有用户登录
If Len(登录用户.用户名) Then                                   '检测是否有用户登录
    If 登录用户.管理用户 = True Then                           '检测用户是否具有管理权限
        权限管理.Show                                          '显示权限管理窗口
    Else
        MsgBox "当前用户没有权限管理用户权限设置！", vbOKOnly + vbInformation '提示无管理权限
    End If
Else
    MsgBox "您尚未登录系统，请首先登录系统！", vbOKOnly + vbInformation        '提示未登录
End If
End Sub

Public Sub 供货商管理窗口()
'在打开供货商管理窗口前，检测当前是否有用户登录并且检查当前用户是否有权限操作供应商资料
If Len(登录用户.用户名) Then                                   '检测用户是否登录
    If 登录用户.供货商资料建立 = True Then                     '检测用户是否具有供货商资料建立权限
        供货商资料管理.Show                                    '显示供货商资料管理权限
    Else
        MsgBox "当前用户没有权限操作供应商资料！", vbOKOnly + vbInformation   '提示无权限
    End If
Else
    MsgBox "您尚未登录系统，请首先登录系统！", vbOKOnly + vbInformation        '提示未登录
End If
End Sub

Public Sub 商品资料管理窗口()
'在打开商品资料管理窗口前，检测当前是否有用户登录并且检查当前用户是否有权限操作商品资料
If Len(登录用户.用户名) Then                                   '检测用户是否登录
    If 登录用户.商品资料建立 = True Then                       '检测用户是否有商品资料建立权限
        商品资料管理.Show                                      '显示商品资料建立窗口
    Else
        MsgBox "当前用户没有权限操作商品资料！", vbOKOnly + vbInformation     '提示无权限
    End If
Else
    MsgBox "您尚未登录系统，请首先登录系统！", vbOKOnly + vbInformation        '提示未登录
End If
End Sub
```

```
Public Sub 进货管理窗口()
'在打开进货管理窗口前，检测当前是否有用户登录并且检查当前用户是否有权限操作进货
If Len(登录用户.用户名) Then                        '检测用户是否登录
    If 登录用户.进货 = True Then                    '检测用户是否有商品资料建立权限
        进货资料管理.Show                          '显示进货资料管理窗口
    Else
        MsgBox "当前用户没有权限操作进货管理！", vbOKOnly + vbInformation    '提示无权限
    End If
Else
    MsgBox "您尚未登录系统，请首先登录系统！", vbOKOnly + vbInformation      '提示未登录
End If
End Sub

Public Sub 销售管理窗口()
'在打开销售管理窗口前，检测当前是否有用户登录并且检查当前用户是否有权限操作销售管理
If Len(登录用户.用户名) Then                        '检测用户是否登录
    If 登录用户.销售 = True Then                    '检测用户是否有销售管理权限
        销售资料管理.Show                          '显示销售资料管理窗口
    Else
        MsgBox "当前用户没有权限操作销售管理！", vbOKOnly + vbInformation    '提示无权限
    End If
Else
    MsgBox "您尚未登录系统，请首先登录系统！", vbOKOnly + vbInformation      '提示未登录
End If
End Sub

Public Sub 销售统计分析窗口()
'在打开销售分析窗口前，检测当前是否有用户登录并且检查当前用户是否有权限操作销售分析
If Len(登录用户.用户名) Then                        '检测用户是否登录
    If 登录用户.销售分析 = True Then                '检测用户是否有销售统计分析管理权限
        销售统计分析.Show                          '显示销售统计分析窗口
    Else
        MsgBox "当前用户没有权限进行销售分析！", vbOKOnly + vbInformation    '提示无权限
    End If
Else
    MsgBox "您尚未登录系统，请首先登录系统！", vbOKOnly + vbInformation      '提示未登录
End If
End Sub

Public Sub 库存资料浏览窗口()
'在打开库存管理窗口前，检测当前是否有用户登录并且检查当前用户是否有权限操作库存
If Len(登录用户.用户名) Then                        '检测用户是否登录
    If 登录用户.库存 = True Then                    '检测用户是否有库存资料管理权限
        库存资料管理.Show                          '显示库存资料管理窗口
    Else
        MsgBox "当前用户没有权限操作库存管理！", vbOKOnly + vbInformation    '提示无权限
    End If
Else
    MsgBox "您尚未登录系统，请首先登录系统！", vbOKOnly + vbInformation      '提示未登录
End If
```

```
End Sub

Public Sub 资料查询与导出窗口()
'在打开资料查询窗口前，检测当前是否有用户登录并且检查当前用户是否有权限操作查询
With 登录用户
    If Len(.用户名) Then                                  '检测用户是否登录
        '检测用户是否具有数据查询与导出权限
        If .供货商资料查询 Or .进货查询 Or .库存查询 Or .商品资料查询 Or .销售查询 Then
            资料查询与导出.Show                           '显示资料查询与导出窗口
        Else
            MsgBox "该用户没有权限进行查询操作！", vbOKOnly + vbInformation   '提示无权限
        End If
    Else
        MsgBox "您尚未登录系统，请首先登录系统！", vbOKOnly + vbInformation   '提示未登录
    End If
End With
End Sub
```

7.4 系统设置功能模块设计

系统设置模块主要完成各项有关系统的设置工作，包括用户名修改、密码修改、权限修改以及产生数据库。在前面的 Access 数据库设计章节已经介绍了产生数据库的过程，本章将讲述该功能模块的其余部分。本系统的登录界面和权限设置时使用的权限设置登录窗体一样，在本章节将会介绍系统登录界面的设计。

7.4.1 系统公共变量

在系统中，有些变量是全局变量，为了便于管理，系统将这些变量统一存放在“公共变量”模块中。这些变量的具体解释如下：

```
'自定义数据类型登录信息，用于保存登录用户的用户信息，包括用户名、密码以及相应的权限设置
Type 登录信息
    用户名 As String
    密码 As String
    供货商资料建立 As Boolean
    供货商资料查询 As Boolean
    商品资料建立 As Boolean
    商品资料查询 As Boolean
    进货 As Boolean
    进货查询 As Boolean
    销售 As Boolean
    销售查询 As Boolean
    销售分析 As Boolean
    库存 As Boolean
    库存查询 As Boolean
    管理用户 As Boolean
```

```
End Type

'登录用户是登录信息自定义数据类型的一个实例，通过登录用户全局变量可以保存登录用户的相关信息
Public 登录用户 As 登录信息
```

7.4.2　用户登录设计

用户登录模块完成系统的登录功能。在没有登录之前，系统中的所有功能都不能使用。登录系统的窗体界面如图 7-11 所示。用户名列表框控件名称为“用户名”，密码文本框名称为“用户密码”。

在该窗口的代码中共包含了 4 个过程，分别是窗口初始化过程、刷新用户列表过程、【确定】按钮单击事件过程和【取消】按钮单击事件过程。其中刷新用户列表过程被窗口初始化事件调用用于初始化窗口。【确定】按钮单击事件用于检验用户名与密码是否正确，当输入正确用户名和密码后，程序保存了该用户的所有信息。如图 7-12 所示的是【确定】按钮的单击事件过程流程图。



图 7-11　系统登录窗口

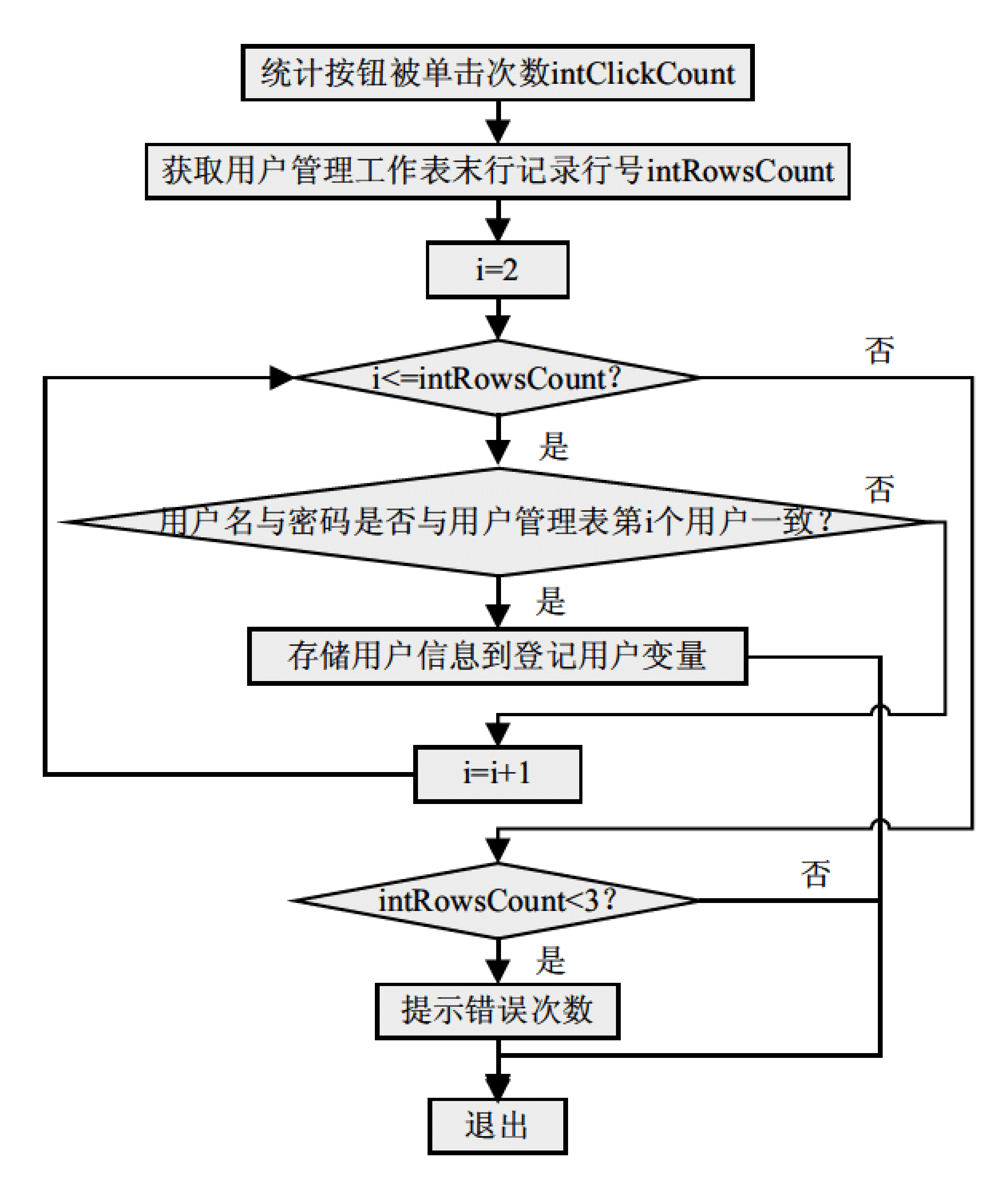


图 7-12　【确定】按钮单击事件过程流程图

登录窗口的详细代码解释如下：

```
Private Sub UserForm_Initialize()
'在窗口初始化时，重置用户名列表
刷新用户名列表
End Sub
```

```
Private Sub 确定_Click()
Dim intRowsCount As Integer, i As Integer
Static intClickCount As Integer                    '该变量记录单击确定按钮次数
intClickCount = intClickCount + 1
intRowsCount = 用户管理.Range("A" & Rows.Count).End(xlUp).Row
'循环检测用户管理表中的用户资料，当找到对应用户后，将用户信息保存倒登录用户公共变量中
For i = 2 To intRowsCount
    If 用户管理.Range("A" & i) = 用户名.Text And 用户管理.Range("B" & i) = 用户密码.Text Then
        With 登录用户
            .用户名 = 用户管理.Range("A" & i)
            .密码 = 用户管理.Range("A" & i)
            .供货商资料建立 = 用户管理.Range("C" & i)
            .供货商资料查询 = 用户管理.Range("D" & i)
            .商品资料建立 = 用户管理.Range("E" & i)
            .商品资料查询 = 用户管理.Range("F" & i)
            .进货 = 用户管理.Range("G" & i)
            .进货查询 = 用户管理.Range("H" & i)
            .销售 = 用户管理.Range("I" & i)
            .销售查询 = 用户管理.Range("J" & i)
            .销售分析 = 用户管理.Range("K" & i)
            .库存 = 用户管理.Range("L" & i)
            .库存查询 = 用户管理.Range("M" & i)
            .管理用户 = 用户管理.Range("N" & i)
        End With
        Unload Me
        Exit Sub
    End If
Next
'当输入用户名或密码错误时，提示错误次数。当错误 3 次后，退出系统
If intClickCount < 3 Then
    MsgBox "用户不存在或用户密码输入错误！已经输入" & intClickCount & "次，你只能尝试 3 次",
vbOKOnly + vbInformation
Else
    ThisWorkbook.Close
End If
End Sub

Private Sub 退出_Click()
Unload Me
End Sub

Sub 刷新用户名列表()
Dim intRowsCount As Integer, i As Integer
intRowsCount = 用户管理.Range("A" & Rows.Count).End(xlUp).Row
Me.用户名.Clear                                  '清空用户名组合框
'重置用户名组合框项目
For i = 2 To intRowsCount
    Me.用户名.AddItem 用户管理.Range("A" & i)
Next
End Sub
```

7.4.3 修改用户名功能设计

修改用户名和修改密码都使用了一个独立的窗口。修改用户名时，需要旧用户名、用户密码、新用户名方可完成修改工作。修改用户名的窗体界面如图 7-13 所示。当前用户如果有管理用户的权限时，可以设置原用户名，否则不能修改原用户名文本框中的内容。

窗口中一共包含了 3 个事件过程的代码。这 3 个过程分别是窗口初始化事件过程、【确定】按钮单击事件过程和【取消】按钮单击事件过程。3 个事件过程中【确定】按钮单击事件过程稍显复杂，这里对该过程做详细介绍。单击【确定】按钮时，程序需要检测输入的用户名是否存在以及密码是否正确。当在用户管理表中找到该用户并且密码正确时，修改该用户的用户名为新用户名即可。如图 7-14 所示的是该过程的流程图。

修改用户名
原用户名：
新用户名：
密码：
确定 取消

图 7-13 修改用户名

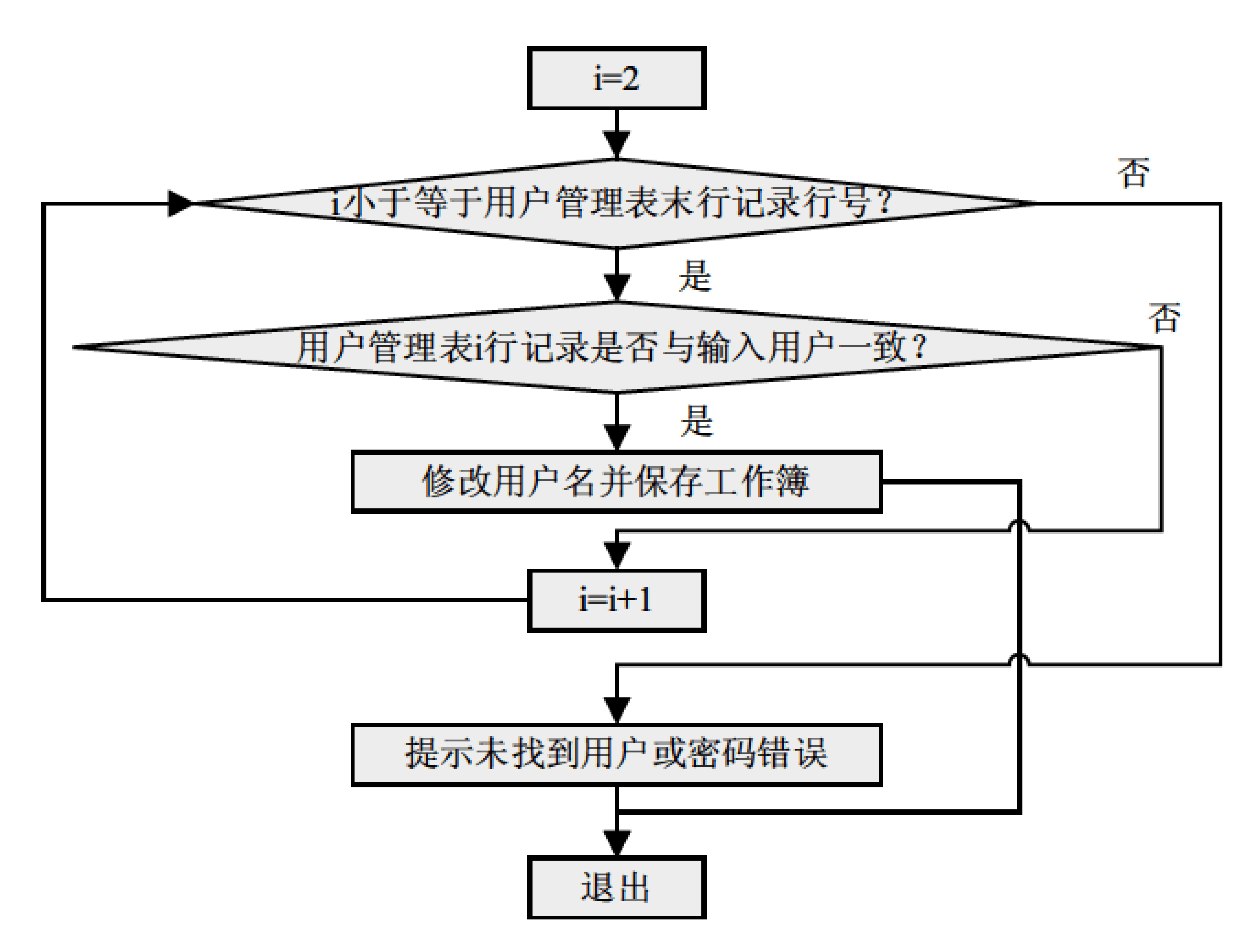


图 7-14 【确定】按钮单击事件过程流程图

该窗体的代码仅包含两个按钮的单击事件代码。详细代码解释如下：

```
Private Sub UserForm_Initialize()
'窗口初始化时，检测登录用户是否有管理用户权限，当没有管理权限时，只能修改自己的名称
If 登录用户.管理用户 = False Then
    原用户名.Text = 登录用户.用户名
    原用户名.Enabled = False
End If
End Sub

Private Sub 确定_Click()
    On Error GoTo errorhandle
    Dim noExist As Boolean
    '在用户管理表中逐个检测用户，当有匹配用户时，修改该用户名并保存工作簿
    For i = 2 To 用户管理.Range("A65536").End(xlUp).Row
        If 用户管理.Range("A" & i).Text = 原用户名.Text And 用户管理.Range("B" & i).Text = 密码
          .Text Then
            用户管理.Range("A" & i) = 新用户名.Text
```

```
                MsgBox "用户名修改成功！请记好您的新用户名！", _
                    vbInformation, "用户名修改成功"
                Unload 修改用户名
                '保存工作簿
                ThisWorkbook.Save
                Exit Sub
            End If
        Next
        MsgBox "没有用户名 " & 原用户名.Text & "或密码错误 ！", vbCritical, "警告"
        Unload 修改用户名
End Sub

Private Sub 取消_Click()
        Unload 修改用户名
End Sub
```

7.4.4 修改密码功能设计

修改密码时，需要用户名、原密码和新密码。修改密码的窗体界面如图 7-15 所示。该窗口和修改用户名窗口类似，代码也包含了 3 个事件过程。其中【确定】按钮单击事件过程稍显复杂，不过该过程的流程和修改用户名窗口的【确定】按钮单击事件流程大体一致。如图 7-16 所示的是该过程的流程图。



图 7-15 修改密码

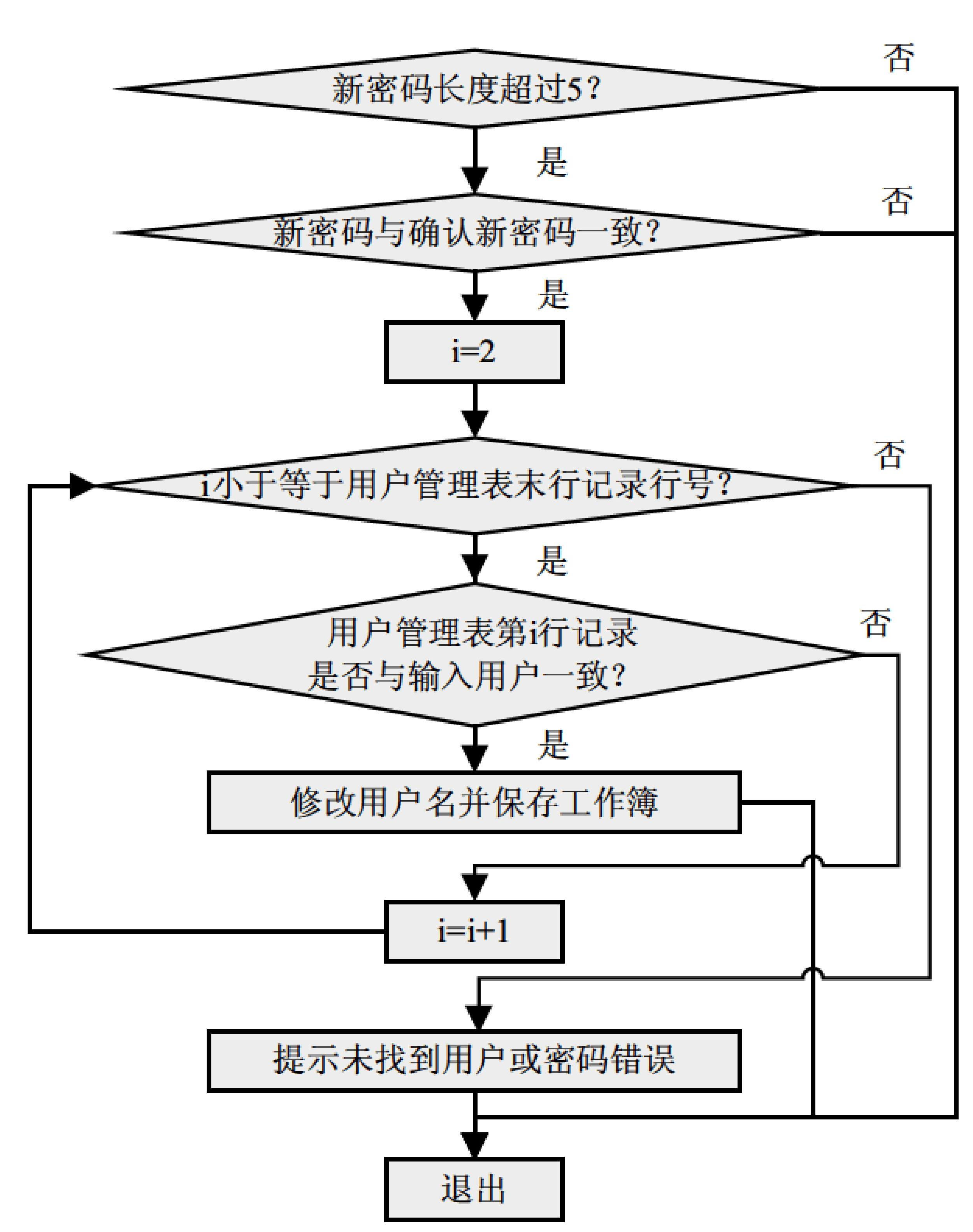


图 7-16 修改密码过程流程

该窗体的代码详细解释如下：

```
Private Sub UserForm_Initialize()
'窗口初始化时，检测登录用户是否有管理用户权限，当没有管理权限时，只能修改自己的密码
If 登录用户.管理用户 = False Then
    用户名.Text = 登录用户.用户名
    用户名.Enabled = False
End If
End Sub

Private Sub CommandButton1_Click()
    On Error GoTo errorhandle
    Dim noExist As Boolean
    '当密码长度少于 5 时，提示重新设置密码，密码长度不能少于 5
    If Len(新密码.Text) < 5 Then
        MsgBox "为安全起见，密码不能小于 5 位！", vbCritical, "注意"
        新密码.Text = ""
        确认新密码.Text = ""
        新密码.SetFocus
        Exit Sub
    ElseIf 新密码.Text <> 确认新密码.Text Then   '当新密码与确认密码不一致时，提示重新设置密码
        MsgBox "两次输入的密码不一致！", vbCritical, "警告"
        新密码.Text = ""
        确认新密码.Text = ""
        Exit Sub
    End If
    '在用户管理表中逐个对照，当有匹配用户时，修改该用户的密码
    For i = 2 To 用户管理.Range("A65536").End(xlUp).Row
        If 用户管理.Range("A" & i).Text = 用户名.Text Then
            用户管理.Range("B" & i) = 新密码.Text
            MsgBox "密码修改成功!请记好您的新密码!", vbInformation, "修改密码"
            Unload 修改密码
            '保存工作簿
            ThisWorkbook.Save
            Exit Sub
        End If
    Next
    MsgBox "没有用户名 " & 用户名.Text & " ！", vbCritical, "警告"
    Unload 修改密码
End Sub

Private Sub CommandButton2_Click()
    Unload 修改密码
End Sub
```

7.4.5　用户权限管理设计

系统对每个用户操作系统的权限进行了细分，分别包括供应商资料、商品资料、进货、

销售、库存的查询和修改权限以及用户管理的权限。这些权限的设置都保存在了用户名密码表中。权限设置的窗体界面如图 7-17 所示。该窗口一共包含了 4 个事件过程，分别是窗口初始化过程、【确定】按钮单击事件过程、【关闭】按钮单击事件过程和用户列表改变事件过程。

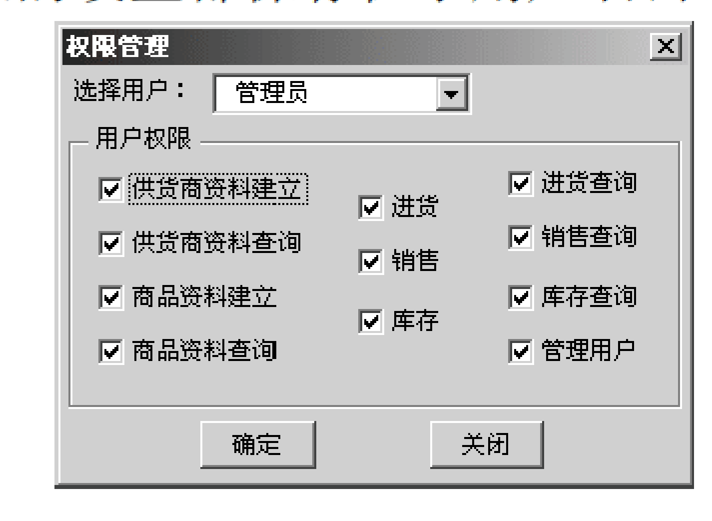


图 7-17　用户权限管理

这些事件代码都不复杂，主要完成一些赋值操作，由于需要赋值的项目比较多，因而其中部分过程占用了比较多的篇幅。以下不再给出这些过程的流程图。

该窗体的详细代码解释如下：

```
Public 行号 As Integer

Private Sub UserForm_Initialize()
Dim intRowsCount As Integer, i As Integer
Application.EnableEvents = False
intRowsCount = 用户管理.Range("A" & Rows.Count).End(xlUp).Row    '获取用户管理末条记录行号
For i = 2 To intRowsCount                                        '循环用户管理所有用户记录
    用户列表.AddItem 用户管理.Range("A" & i)                      '为用户列表添加项目
Next
用户列表.ListIndex = 0                                            '设置用户列表默认选定项
Application.EnableEvents = True
End Sub

Private Sub 关闭_Click()
Unload Me                                                        '关闭窗口
End Sub

Private Sub 确定_Click()
With 用户管理
    .Range("C" & 行号) = 供货商资料建立.Value                      '设置用户供货商资料建立权限
    .Range("D" & 行号) = 供货商资料查询.Value                      '设置用户供货商资料查询权限
    .Range("E" & 行号) = 商品资料建立.Value                        '设置用户商品资料建立权限
    .Range("F" & 行号) = 商品资料查询.Value                        '设置用户商品资料查询权限
    .Range("G" & 行号) = 进货.Value                                '设置用户进货权限
    .Range("H" & 行号) = 进货查询.Value                            '设置用户进货查询权限
    .Range("I" & 行号) = 销售.Value                                '设置用户销售权限
    .Range("J" & 行号) = 销售查询.Value                            '设置用户销售查询权限
    .Range("K" & 行号) = 销售分析.Value                            '设置用户销售分析权限
    .Range("L" & 行号) = 库存.Value                                '设置用户库存权限
    .Range("M" & 行号) = 库存查询.Value                            '设置用户库存查询权限
    .Range("N" & 行号) = 管理用户.Value                            '设置管理用户权限
End With
Unload Me
End Sub

Private Sub 用户列表_change()
Dim intRowsCount As Integer, i As Integer
```

```
intRowsCount = 用户管理.Range("A" & Rows.Count).End(xlUp).Row      '获取用户管理末条记录行号
For i = 2 To intRowsCount                                         '循环用户管理所有用户记录
    If 用户管理.Range("A" & i) = 用户列表.Text Then                '检测用户管理表第 i 个用户名是否与选
                                                                   定用户一致
        With 用户管理
            供货商资料建立.Value = CBool(.Range("C" & i))           '设置供货商资料建立单选按钮
            供货商资料查询.Value = CBool(.Range("D" & i))           '设置供货商资料查询单选按钮
            商品资料建立.Value = CBool(.Range("E" & i))             '设置商品资料建立单选按钮
            商品资料查询.Value = CBool(.Range("F" & i))             '设置商品资料查询单选按钮
            进货.Value = CBool(.Range("G" & i))                     '设置进货单选按钮
            进货查询.Value = CBool(.Range("H" & i))                 '设置进货查询单选按钮
            销售.Value = CBool(.Range("I" & i))                     '设置销售单选按钮
            销售查询.Value = CBool(.Range("J" & i))                 '设置销售查询单选按钮
            销售分析.Value = CBool(.Range("K" & i))                 '设置销售分析单选按钮
            库存.Value = CBool(.Range("L" & i))                     '设置库存单选按钮
            库存查询.Value = CBool(.Range("M" & i))                 '设置库存查询单选按钮
            管理用户.Value = CBool(.Range("N" & i))                 '设置管理用户单选按钮
        End With
        Exit Sub
    End If
        供货商资料建立.Value = False                                '取消供货商资料单选按钮选定
        供货商资料查询.Value = False                                '取消供货商资料查询单选按钮选定
        商品资料建立.Value = False                                  '取消商品资料建立单选按钮选定
        商品资料查询.Value = False                                  '取消商品资料查询单选按钮选定
        进货.Value = False                                          '取消进货单选按钮选定
        进货查询.Value = False                                      '取消近乎偶查询单选按钮选定
        销售.Value = False                                          '取消销售单选按钮选定
        销售查询.Value = False                                      '取消销售查询单选按钮选定
        销售分析.Value = False                                      '取消销售查询单选按钮选定
        库存.Value = False                                          '取消库存单选按钮选定
        库存查询.Value = False                                      '取消库存查询单选按钮选定
        管理用户.Value = False                                      '取消管理用户单选按钮选定
Next
End Sub
```

7.5　供应商资料管理窗体设计

7.5.1　供应商资料管理窗体界面

每一个供应商，可能有的资料信息包括供应商名称、通讯地址、供应商编码、邮政编码、联系电话、传真号码、联系人、联系人电话、联系人 Email 以及备注项目。对于供应商资料，通常需要完成新建、保存、修改、删除和查询工作，这些都可以通过供应商资料管理窗体完成。该窗口界面如图 7-18 所示。

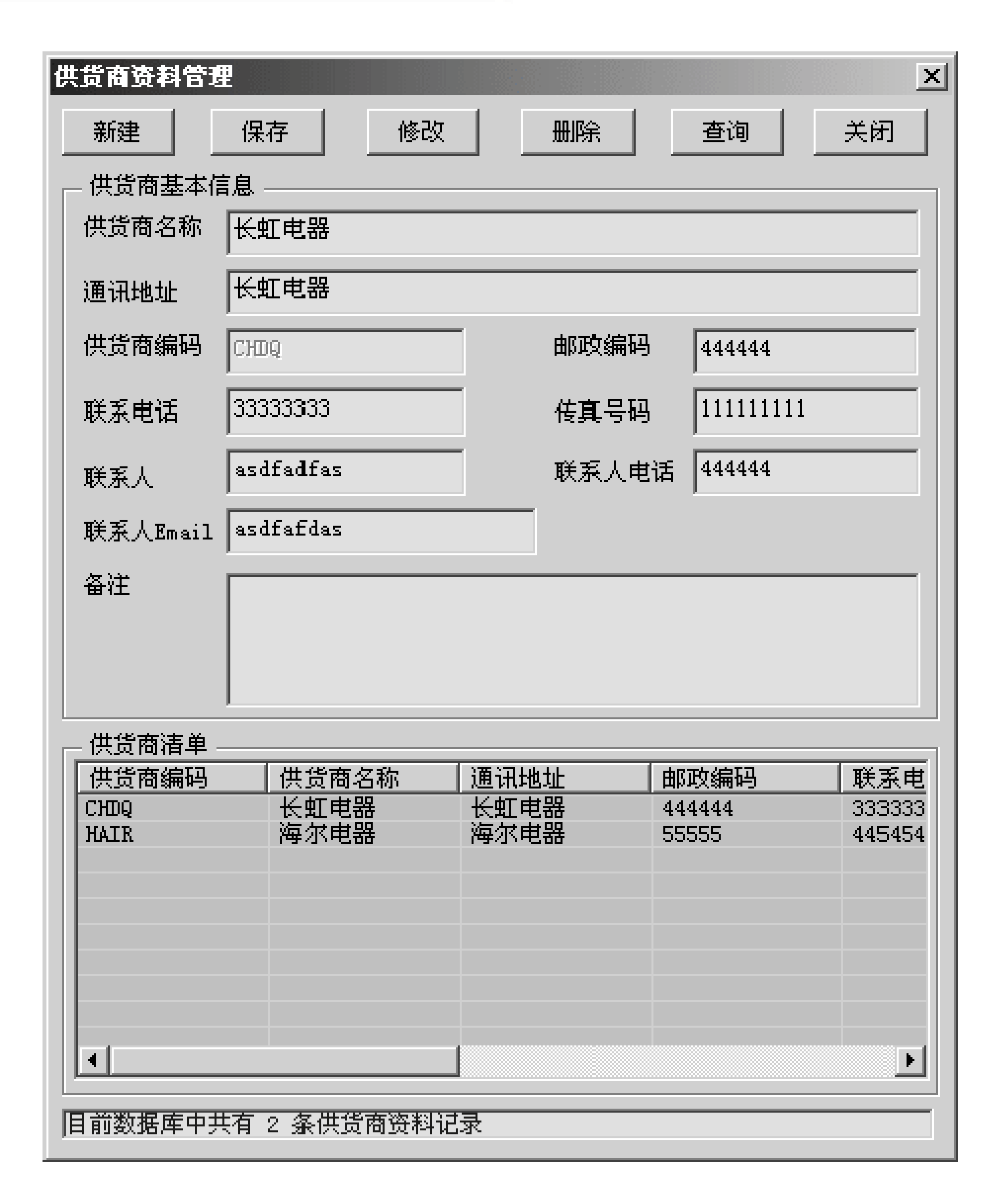


图 7-18　供应商资料管理界面

7.5.2　窗口初始化与关闭事件代码设计

窗口初始化时，需要初始化窗口中使用到的数组 myArray、建立到数据库的链接以及刷新窗口显示。刷新窗口通过 3 个过程完成：查询供货商信息过程获取供货商信息记录集，显示供货商信息过程将第一条记录数据显示在窗口的各个文本框中，myListView 过程将所有供货商信息显示在供货商清单中。

以下是该窗体的初始化过程与关闭事件详细代码解释：

```
Dim myArray As Variant
Dim cnn As New ADODB.Connection
Dim rs As ADODB.Recordset

Private Sub UserForm_Initialize()
    Dim i As Integer
    Dim SQL As String
    myArray = Array("供货商编码", "供货商名称", "通讯地址", "邮政编码", "联系电话", _
        "传真号码", "联系人", "联系人电话", "联系人 Email", "备注")
    '建立与数据库的链接
    With cnn
        .ConnectionString = "Provider=microsoft.jet.oledb.4.0;" _
        & "Data Source=" & ThisWorkbook.Path & "\进销存数据库.mdb;" _
        & "Jet Oledb:database password=123456;"
        .Open
    End With
    查询供货商信息
    显示供货商信息
    myListView
```

```
End Sub

Private Sub 关闭退出_Click()
    cnn.Close
    Set rs = Nothing
    Set cnn = Nothing
    Unload 供货商资料管理
End Sub
```

7.5.3　保存按钮单击事件代码设计

【保存】按钮用于将用户新建的供货商资料保存到数据库中。该按钮单击事件过程首先检测用户是否输入了必要的供货商数据，然后确认供货商编号与数据库已有编号是否重复，随后程序还需要确认各个数据的长度不超过数据库允许长度，最后程序将供货商资料保存到数据库中并刷新窗口显示。如图 7-19 所示的是该过程的流程图。

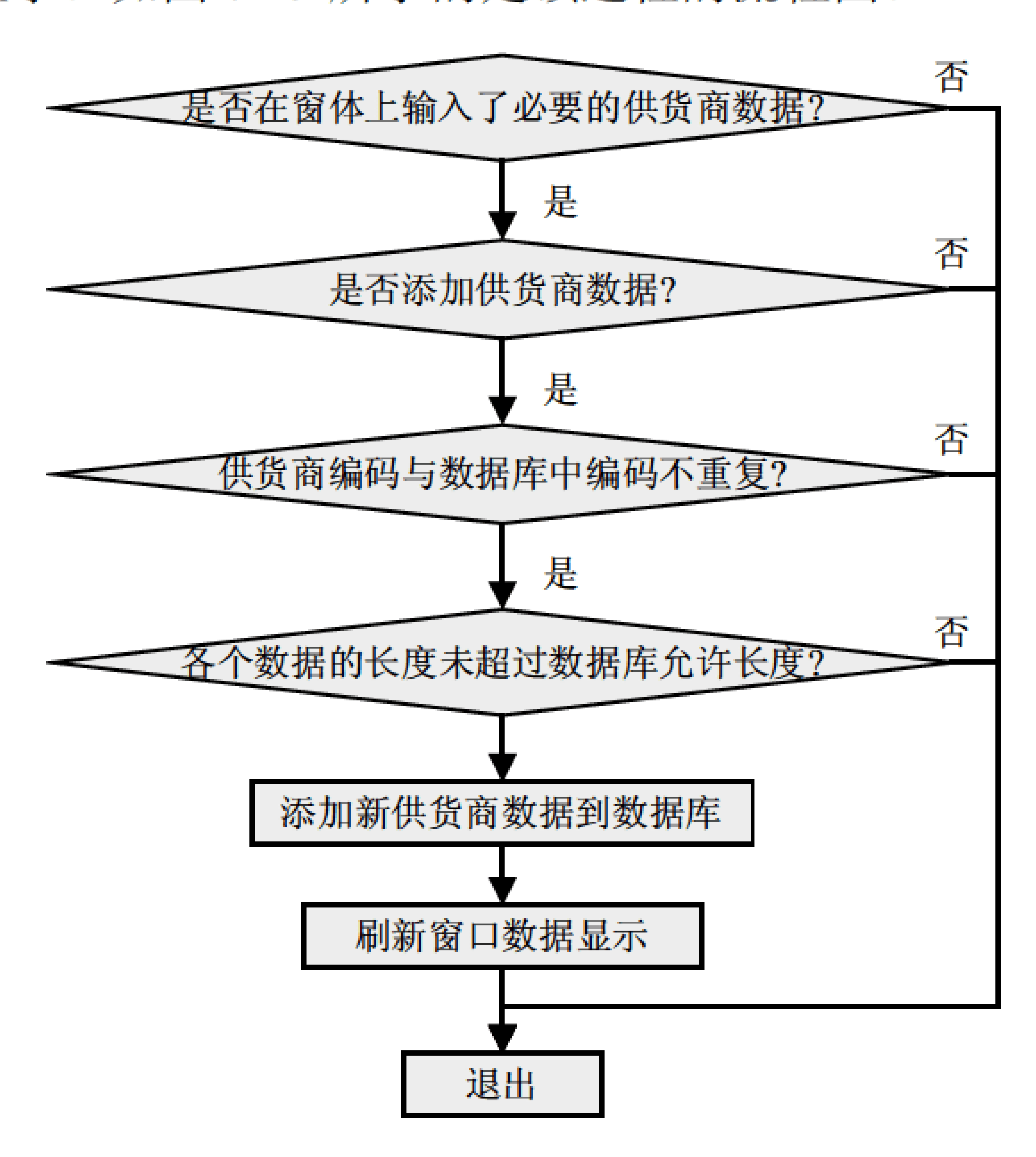


图 7-19　【保存】按钮单击事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 保存记录_Click()
    Dim i As Integer
    '判断是否在窗体上输入了必要的供货商数据
    For i = 0 To UBound(myArray) -1
        If Me.Controls(myArray(i)).Name <> "备注" Then
            If Me.Controls(myArray(i)).Value = "" Then
                MsgBox Me.Controls(myArray(i)).Name & "不能为空！", vbCritical
                Me.Controls(myArray(i)).SetFocus
                Exit Sub
```

```
                End If
            End If
        Next
        '首先判断在数据库中是否存在相同的供货商编码
        Dim rsNum As New ADODB.Recordset
        SQL = "select 供货商编码 from 供货商信息 " _
            & "where 供货商编码='" & 供货商编码.Value & "'"
        rsNum.Open SQL, cnn, adOpenKeyset, adLockOptimistic
        If rsNum.BOF = False And rsNum.EOF = False Then
            MsgBox "在数据库中已经存在有编号为<" & 供货商编码.Value _
                & ">的供货商记录！" & vbCrLf _
                & "请重新输入供货商编码！", vbOKOnly + vbCritical
            Me.供货商编码.Value = ""
            Me.供货商编码.SetFocus
            GoTo hhh
        End If
        '准备将窗体上的数据添加到数据库中
        SQL = "select * from 供货商信息"
        Set rs = New ADODB.Recordset
        rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
        '判断各个数据的长度是否超过了数据库允许的长度
        For i = 0 To UBound(myArray)
            If Len(Me.Controls(myArray(i)).Value) > rs.Fields(i).DefinedSize Then
                MsgBox Me.Controls(myArray(i)).Name _
                    & "的数据长度已经超过了数据库规定的长度！", vbCritical
                Me.Controls(myArray(i)).Value = Left(Me.Controls(myArray(i)).Value, _
                    rs.Fields(i).DefinedSize)
                Me.Controls(myArray(i)).SetFocus
                Exit Sub
            End If
        Next
        If MsgBox("本操作将添加新的供货商记录！" & vbCrLf & "是否要添加？", _
            vbQuestion + vbYesNo, "添加记录") = vbNo Then Exit Sub
        '开始添加数据
        With rs
            .AddNew                        '添加各个字段的数据
            For i = 0 To UBound(myArray)
                .Fields(i) = Me.Controls(myArray(i)).Value
            Next
            .Update                        '更新数据表
        End With
        MsgBox "已经成功将新供货商数据添加到数据库中！", vbInformation, "添加记录"
        '刷新查询和显示
        查询供货商信息
        显示供货商信息
        myListView
hhh:
        rsNum.Close
        Set rsNum = Nothing
End Sub
```

7.5.4　新建按钮单击事件代码设计

【新建】按钮用于重设窗口中所有输入框，便于用户输入新供应商信息。窗口中所有的输入控件的名称在窗口初始化过程中已经保存，程序通过一个 For 循环遍历窗口所有输入控件。新建按钮的详细代码解释如下：

```
Private Sub 新建记录_Click()
    For i = 0 To UBound(myArray)                        '遍历窗口所有控件
        Me.Controls(myArray(i)).Value = ""              '置空输入控件
    Next
    供货商编码.Enabled = True                           '设置供应商编码控件可用
    供货商编码.SetFocus                                 '定位输入焦点到供应商编码控件
End Sub
```

7.5.5　修改按钮单击事件代码设计

【修改】按钮将用户对供应商资料做出的修改操作保存到数据库中。在确认用户需要修改数据后，程序通过一个更新查询 SQL 语句完成修改数据库记录操作，最后程序通过调用查询供货商信息、显示供货商信息与 myListView 过程刷新窗口的数据显示。如图 7-20 所示的是该过程的流程图。

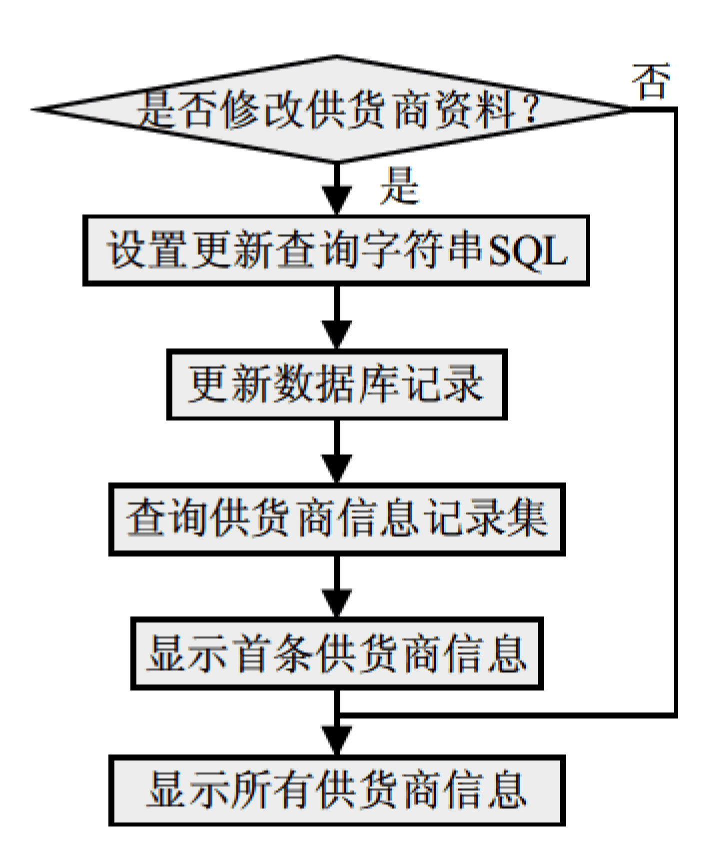


图 7-20　【修改】按钮单击事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 修改记录_Click()
    If MsgBox("本操作将修改编码为<" & 供货商编码.Value & ">的供货商信息记录！" _
        & vbCrLf & "是否要修改？", _
        vbQuestion + vbYesNo + vbDefaultButton2, "修改记录") = vbNo Then Exit Sub
    '修改更新记录
    SQL = "update 供货商信息 set " _
        & "供货商名称='" & 供货商名称.Value & "'," _
```

```
            & "通讯地址='" & 通讯地址.Value & "'," _
            & "邮政编码='" & 邮政编码.Value & "'," _
            & "联系电话='" & 联系电话.Value & "'," _
            & "传真号码='" & 传真号码.Value & "'," _
            & "联系人='" & 联系人.Value & "'," _
            & "联系人电话='" & 联系人电话.Value & "'," _
            & "联系人 Email='" & 联系人 Email.Value & "'," _
            & "备注='" & 备注.Value & "' " _
            & "where 供货商编码='" & 供货商编码.Value & "'"
        Set rs = New ADODB.Recordset
        rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
        MsgBox "已经成功将编码为<" & 供货商编码.Value _
            & ">的供货商信息记录进行修改！", vbInformation, "修改记录"
        '刷新查询和显示
        查询供货商信息
        显示供货商信息
        myListView
End Sub
```

7.5.6 删除按钮单击事件代码设计

【删除】按钮用于删除当前显示的供应商资料记录，程序通过一个删除查询在数据库中完成删除操作。在删除操作完成之后，程序刷新了窗口显示的记录数据。以下是该过程的详细代码解释：

```
Private Sub 删除记录_Click()
        If MsgBox("本操作将删除编码为<" & 供货商编码.Value & ">的供货商信息记录！" _
            & vbCrLf & "是否要删除？", _
            vbQuestion + vbYesNo + vbDefaultButton2, "删除记录") = vbNo Then Exit Sub
        SQL = "delete from 供货商信息 where 供货商编码='" & 供货商编码.Value & "'"
        Set rs = cnn.Execute(SQL)
        MsgBox "已经成功将编码为<" & 供货商编码.Value & ">的供货商信息记录删除！", _
            vbInformation, "删除记录"
        '刷新查询和显示
        查询供货商信息
        显示供货商信息
        myListView
End Sub
```

7.5.7 查询按钮单击事件代码设计

在供应商的查询操作中只能按照供货商编码进行查询。单击【查询】按钮后，弹出一个供货商编码输入框。用户输入完供应商编码后，程序从数据库中获取供货商编码为用户输入结果的所有供货商信息，并将查询到的记录信息显示到窗口中。如图 7-21 所示的是该过程的流程图。

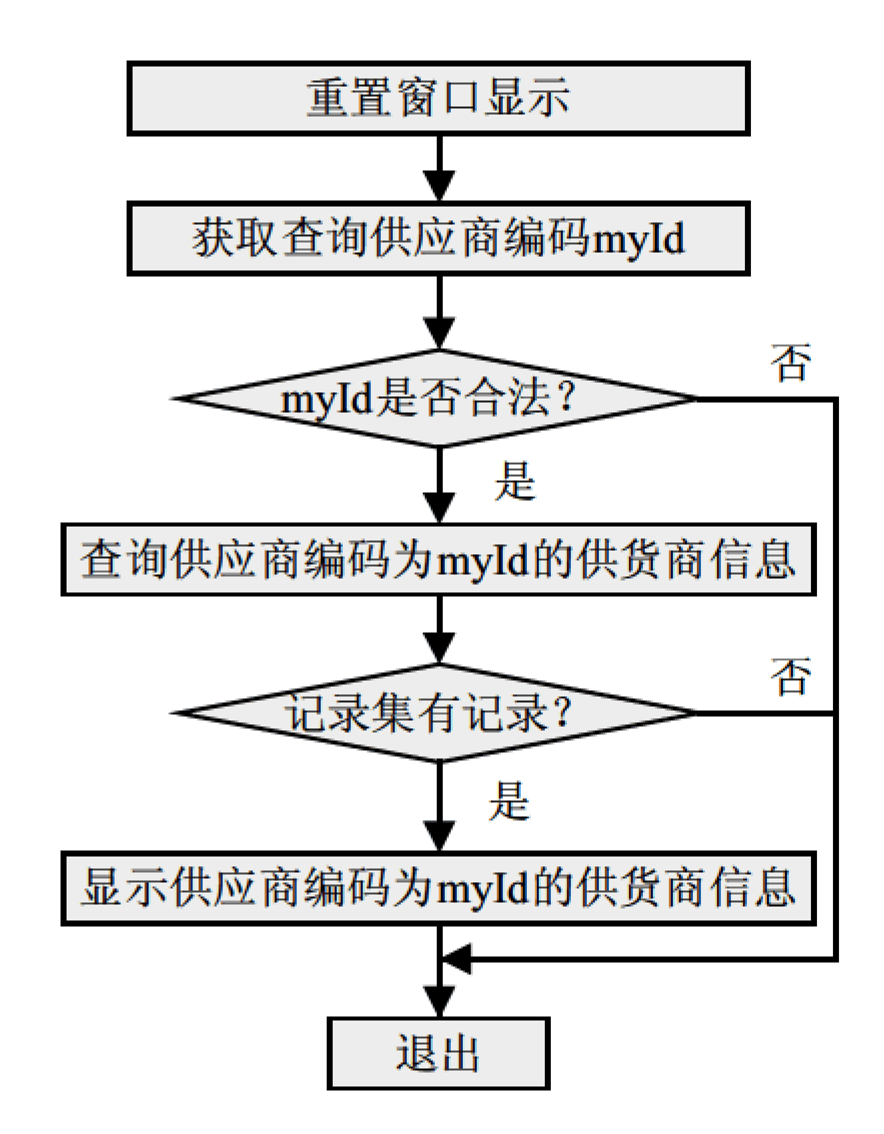


图 7-21　【查询】按钮单击事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 查询记录_Click()
    Dim myId As String
    Dim SQL As String
    Dim i As Integer
    For i = 0 To UBound(myArray)
        Me.Controls(myArray(i)).Value = ""
    Next
    myId = InputBox("请输入供货商编码：", "供货商查询")
    If Len(Trim(myId)) = 0 Then
        MsgBox "没有输入供货商编码！", vbCritical, "警告"
        Exit Sub
    End If
    SQL = "select * from 供货商信息 where 供货商编码='" & myId & "'"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If rs.BOF And rs.EOF Then
        MsgBox "没有编码为<" & myId & ">的供货商信息!", vbCritical, "查询结果"
    Else
        显示供货商信息
        供货商编码.Enabled = False
    End If
End Sub
```

7.5.8　ListView 控件项目单击事件代码设计

用户在 ListView 控件中单击某个供应商项目后，需要将该项目的所有数据显示到窗口上部对应的文本框中。程序首先将各个文本框的值清空，然后通过 For 循环，使用 ListView 控件的 ListItems 属性定位项目里各个子项目，并把这些子项的数据显示到窗口上面对应的文本

框中。以下是该过程的详细代码解释：

```
Private Sub ListView1_ItemClick(ByVal Item As MSComctlLib.ListItem)
    On Error Resume Next
    Dim i As Integer
    供货商编码.Enabled = False
    For i = 0 To UBound(myArray)
        Me.Controls(myArray(i)).Value = ""
    Next
    Me.Controls(myArray(0)).Value = ListView1.ListItems(Item.Index)
    For i = 1 To UBound(myArray)
        Me.Controls(myArray(i)).Value = ListView1.ListItems(Item.Index).SubItems(i)
    Next i
End Sub
```

7.5.9 查询与显示供货商信息过程代码设计

查询与显示供货商信息两个过程共同完成查询记录集并将记录集数据显示到窗口的工作。查询供货商信息首先生成一个查询供货商信息字符串，然后按照该查询字符串打开记录集，从而获取对应查询信息。显示供货商信息过程使用在查询供货商信息过程中获取的供货商信息记录集，将该记录集首条记录数据写入到窗口的各个对应文本框中。以下是这两个过程的详细代码解释：

```
Public Sub 查询供货商信息()
    SQL = "select * from 供货商信息 order by 供货商编码"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
End Sub

Public Sub 显示供货商信息()
    On Error Resume Next
    Dim i As Integer
    新建记录_Click
    '显示第一个供货商信息
    rs.MoveFirst
    For i = 0 To UBound(myArray)
        If IsNull(rs.Fields(i)) Then
            Me.Controls(myArray(i)).Value = ""
        Else
            Me.Controls(myArray(i)).Value = rs.Fields(i)
        End If
    Next
End Sub
```

7.5.10 myListView 过程代码设计

myListView 过程完成 ListView 控件的显示设置以及标题任务，然后程序将 rs 记录集的所

有记录数据显示到控件上。如图 7-22 所示的是该过程流程图。

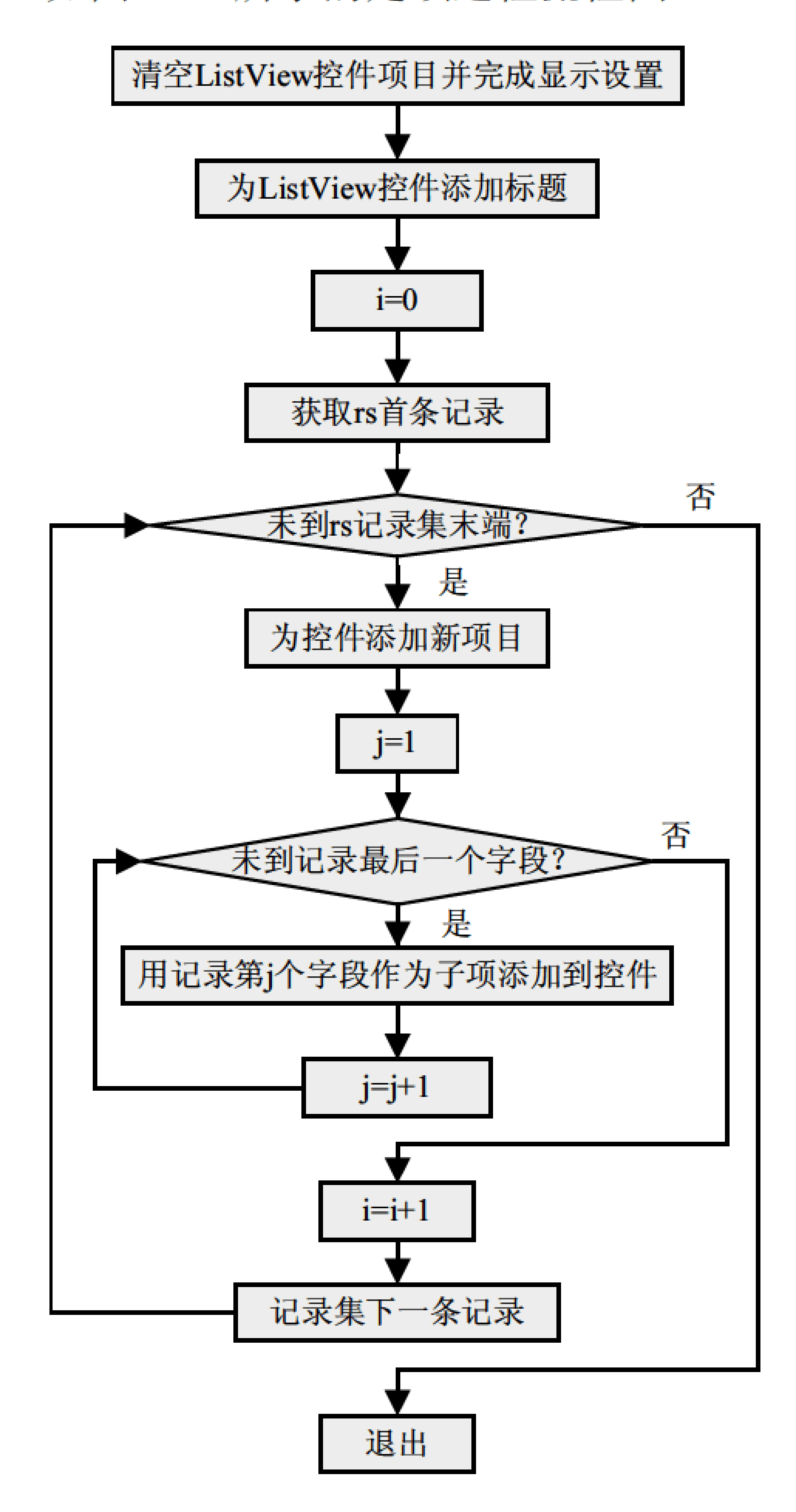


图 7-22　myListView 过程流程图

以下是该过程的详细代码解释：

```
Private Sub myListView()
    On Error Resume Next
    Dim i As Integer
    '设置 ListView 的标题
    With ListView1
        .ColumnHeaders.Clear
        .ListItems.Clear
        .View = lvwReport
        .FullRowSelect = True
        .Gridlines = True
        .Sorted = True
        .ColumnHeaders.Add , , myArray(0)
        For i = 1 To UBound(myArray)
            .ColumnHeaders.Add , , myArray(i)
        Next
```

```
        '设置 ListView 的各行数据
        i = 0
        rs.MoveFirst
        Do While Not rs.EOF
            .ListItems.Add , , rs.Fields(0).Value
            For j = 1 To rs.Fields.Count -1
                .ListItems(i + 1).SubItems(j) = rs.Fields(j).Value
            Next
            rs.MoveNext
            i = i + 1
        Loop
    End With
    rs.MoveFirst
    供货商数目.Caption = "目前数据库中共有 " & i & " 条供货商资料记录"
End Sub
```

7.6 商品资料管理窗体设计

商品资料管理窗口用于完成商品资料信息的管理工作，通过该窗口用户可以新建、查看、编辑、删除商品基本信息数据。进货模块与销售模块都将会使用到该部分建立的数据。

7.6.1 商品资料管理窗口界面设计

商品资料管理窗口和供应商资料管理窗口相似，整个结构上大体一致，仅仅是商品资料管理相关的项目不一样而已。商品资料管理需要的相关资料包括商品名称、商品编码、商品规格、计量单位、最高库存、最低库存以及备注项目。同供应商资料管理模块一样，该模块也需要完成新建、保存、修改和删除操作。该窗口的界面如图 7-23 所示。

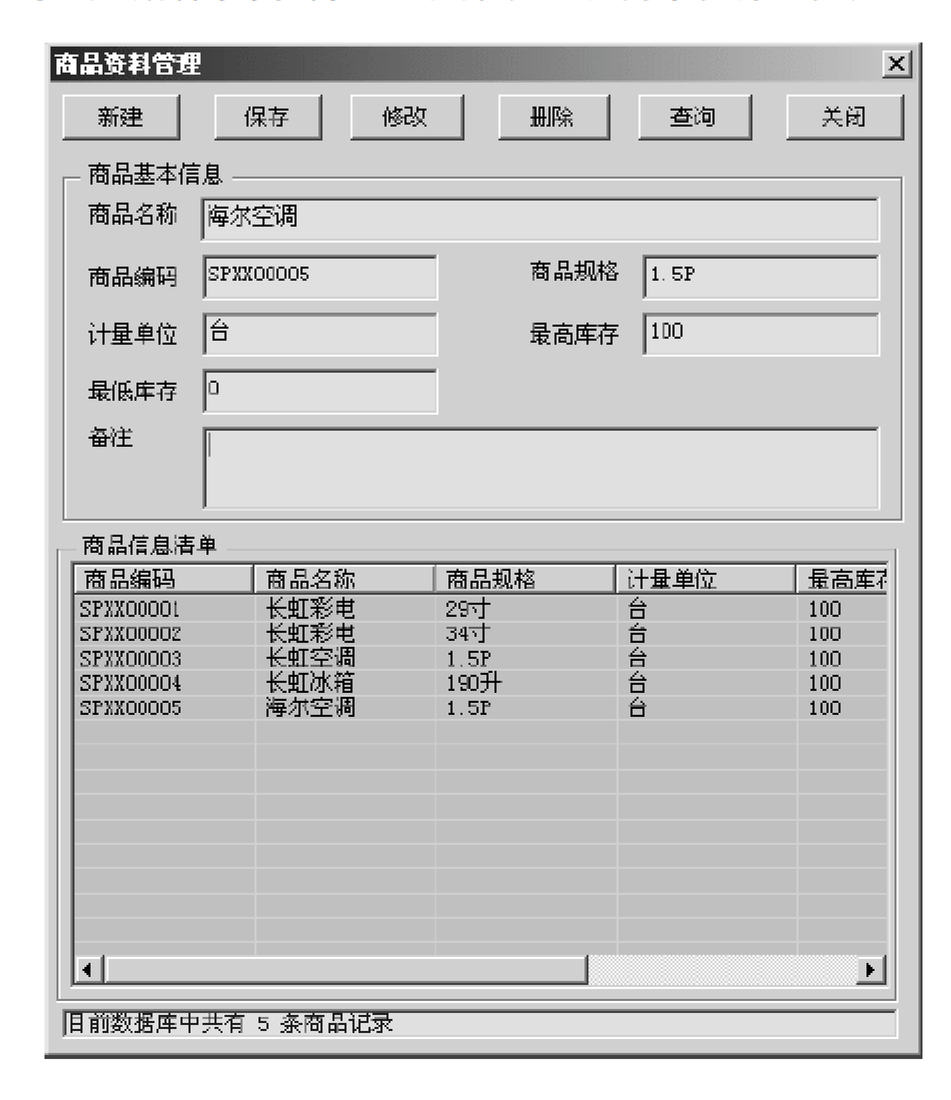


图 7-23 商品资料管理界面

7.6.2　窗口初始化与关闭事件代码设计

窗口初始化时，需要初始化窗口中使用到的数组 myArray、建立到数据库的链接以及刷新窗口显示。刷新窗口通过 3 个过程完成：查询商品信息过程获取商品信息记录集，显示商品信息过程将第一条记录数据显示在窗口的各个文本框中，myListView 过程将所有商品信息显示在商品信息清单中。

以下是该窗体的初始化过程与关闭事件详细代码解释：

```
Dim myArray As Variant
Dim cnn As New ADODB.Connection
Dim rs As ADODB.Recordset

Private Sub UserForm_Initialize()
    Dim i As Integer
    Dim SQL As String
    myArray = Array("商品编码", "商品名称", "商品规格", "计量单位", _
        "最高库存", "最低库存", "备注")
    '建立与数据库的连接
    With cnn
        .ConnectionString = "Provider=microsoft.jet.oledb.4.0;" _
        & "Data Source=" & ThisWorkbook.Path & "\进销存数据库.mdb;" _
        & "Jet Oledb:database password=123456;"
        .Open
    End With
    查询商品信息
    显示商品信息
    myListView
End Sub

Private Sub 关闭退出_Click()
    cnn.Close
    Set rs = Nothing
    Set cnn = Nothing
    Unload 商品资料管理
End Sub
```

7.6.3　保存按钮单击事件代码设计

【保存】按钮用于将用户新建的商品资料保存到数据库中。该按钮单击事件过程首先检测用户是否输入了必要的商品数据，然后确认商品编号与数据库已有编号是否重复，随后程序还需要确认各个数据的长度不超过数据库允许长度，最后程序将商品资料保存到数据库中并刷新窗口显示。如图 7-24 所示的是该过程的流程图。

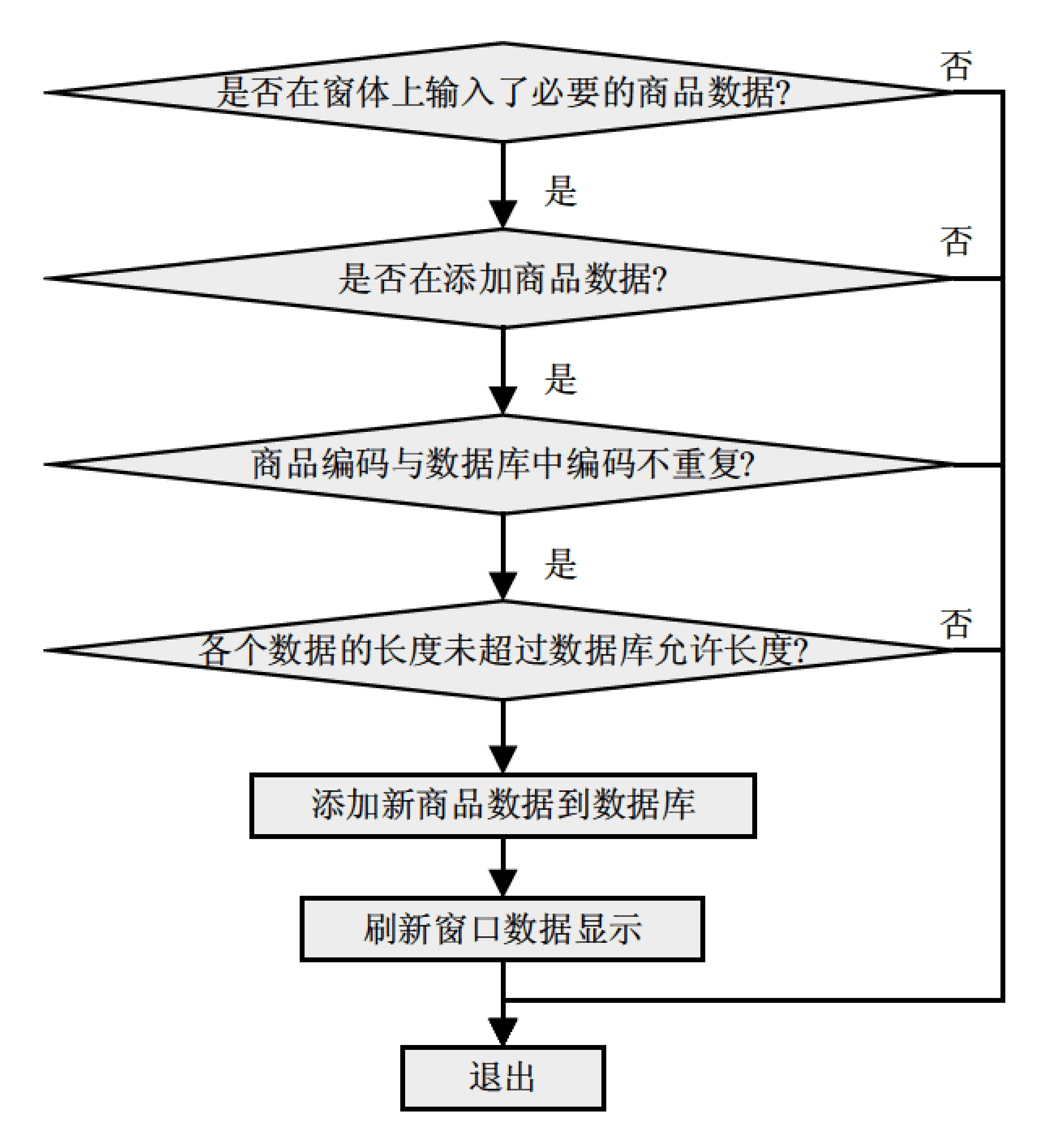


图 7-24 【保存】按钮单击事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 保存记录_Click()
    On Error GoTo xxx
    Dim i As Integer
    '判断是否在窗体上输入了必要的商品数据
    For i = 0 To UBound(myArray) -1
        If Me.Controls(myArray(i)).Name <> "备注" Then
            If Me.Controls(myArray(i)).Value = "" Then
                MsgBox Me.Controls(myArray(i)).Name & "不能为空！", vbCritical
                Me.Controls(myArray(i)).SetFocus
                Exit Sub
            End If
        End If
    Next i
    If MsgBox("本操作将添加新的商品记录！" & vbCrLf & "是否要添加？", _
        vbQuestion + vbYesNo, "添加记录") = vbNo Then Exit Sub
    '首先判断在数据库中是否存在相同的商品编码
    Dim rsNum As New ADODB.Recordset
    SQL = "select 商品编码 from 商品信息 where 商品编码='" & 商品编码.Value & "'"
    rsNum.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If rsNum.BOF = False And rsNum.EOF = False Then
        MsgBox "在数据库中已经存在有编号为<" & 商品编码.Value & ">的商品记录！" _
            & vbCrLf & "请重新输入商品编码！", vbOKOnly + vbCritical
        Me.商品编码.Value = ""
        Me.商品编码.SetFocus
```

```
            Exit Sub
        End If
        '准备将窗体上的数据添加到数据库中
        SQL = "select * from  商品信息"
        Set rs = New ADODB.Recordset
        rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
        '判断各个数据的长度是否超过了数据库允许的长度
        For i = 0 To UBound(myArray)
            If Len(Me.Controls(myArray(i)).Value) > rs.Fields(i).DefinedSize Then
                MsgBox Me.Controls(myArray(i)).Name _
                    & "的数据长度已经超过了数据库规定的长度！", vbCritical
                Me.Controls(myArray(i)).Value = Left(Me.Controls(myArray(i)).Value, _
                    rs.Fields(i).DefinedSize)
                Me.Controls(myArray(i)).SetFocus
                Exit Sub
            End If
        Next i
        '开始添加数据
        With rs
            .AddNew
            For i = 0 To UBound(myArray)
                .Fields(i) = Me.Controls(myArray(i)).Value
            Next i
            .Update
        End With
        MsgBox "已经成功将新商品数据添加到数据库中！", vbInformation, "添加记录"
        '刷新查询和显示
        Call  查询商品信息
        Call  显示商品信息
        Call myListView
hhh:
        rsNum.Close
        Set rsNum = Nothing
        Exit Sub
xxx:
        MsgBox Err.Description, vbCritical, "错误"
End Sub
```

7.6.4　新建按钮单击事件代码设计

【新建】按钮用于重设窗口中所有输入框，便于用户输入新商品信息。窗口中所有的输入控件的名称在窗口初始化过程中已经保存，程序通过一个 For 循环遍历窗口所有输入控件。以下是该过程的详细代码解释：

```
Private Sub  新建记录_Click()
        For i = 0 To UBound(myArray)
            Me.Controls(myArray(i)).Value = ""
        Next i
```

```
    商品编码.Enabled = True
    商品编码.SetFocus
End Sub
```

7.6.5 修改按钮单击事件代码设计

修改按钮用于将用户对商品资料做出的修改操作保存到数据库中。在确认用户需要修改数据后，程序通过一个更新查询 SQL 语句完成修改数据库记录操作，然后通过调用查询商品信息、显示商品信息与 myListView 过程刷新窗口的数据显示。如图 7-25 所示是该过程的流程图。

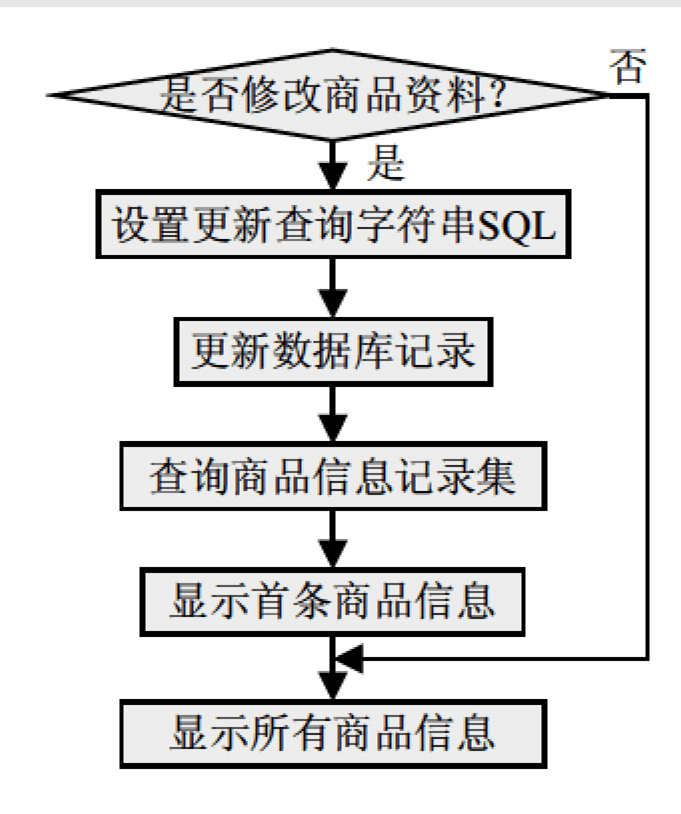


图 7-25 【修改】按钮单击事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 修改记录_Click()
    If MsgBox("本操作将修改编码为<" & 商品编码.Value & ">的商品信息记录！" _
        & vbCrLf & "是否要修改？", _
        vbQuestion + vbYesNo + vbDefaultButton2, "修改记录") = vbNo Then Exit Sub
    '修改更新记录
    SQL = "update 商品信息 set " _
        & "商品名称='" & 商品名称.Value & "'," _
        & "商品规格='" & 商品规格.Value & "'," _
        & "计量单位='" & 计量单位.Value & "'," _
        & "最高库存=" & 最高库存.Value & "," _
        & "最低库存=" & 最低库存.Value & "," _
        & "备注='" & 备注.Value & "' " _
        & "where 商品编码='" & 商品编码.Value & "'"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    MsgBox "已经成功将编码为<" & 商品编码.Value & ">的商品信息记录进行修改！", _
        vbInformation, "修改记录"
    '刷新查询和显示
    Call 查询商品信息
    Call 显示商品信息
    Call myListView
End Sub
```

7.6.6 删除按钮单击事件代码设计

【删除】按钮用于删除当前显示的商品资料记录，程序通过一个删除查询在数据库中完成删除操作。在删除操作完成之后，程序刷新了窗口显示的记录数据。以下是该过程的详细代码解释：

```
Private Sub 删除记录_Click()
    If MsgBox("本操作将删除编码为<" & 商品编码.Value & ">的商品信息记录！" _
```

```
        & vbCrLf & "是否要删除？", _
        vbQuestion + vbYesNo + vbDefaultButton2, "删除记录") = vbNo Then Exit Sub
    SQL = "delete from 商品信息 where 商品编码='" & 商品编码.Value & "'"
    Set rs = cnn.Execute(SQL)
    MsgBox "已经成功将编码为<" & 商品编码.Value & ">的商品信息记录删除！", _
        vbInformation, "删除记录"
    '刷新查询和显示
    Call 查询商品信息
    Call 显示商品信息
    Call myListView
End Sub
```

7.6.7　查询按钮单击事件代码设计

在商品的查询操作中，只能按照商品编码进行查询。单击【查询】按钮后，弹出一个商品编码输入框。用户输入完商品编码后，程序从数据库中获取商品编码为用户所输入结果的所有商品信息，并将查询到的记录信息显示到窗口中。如图 7-26 所示是该过程的流程图：

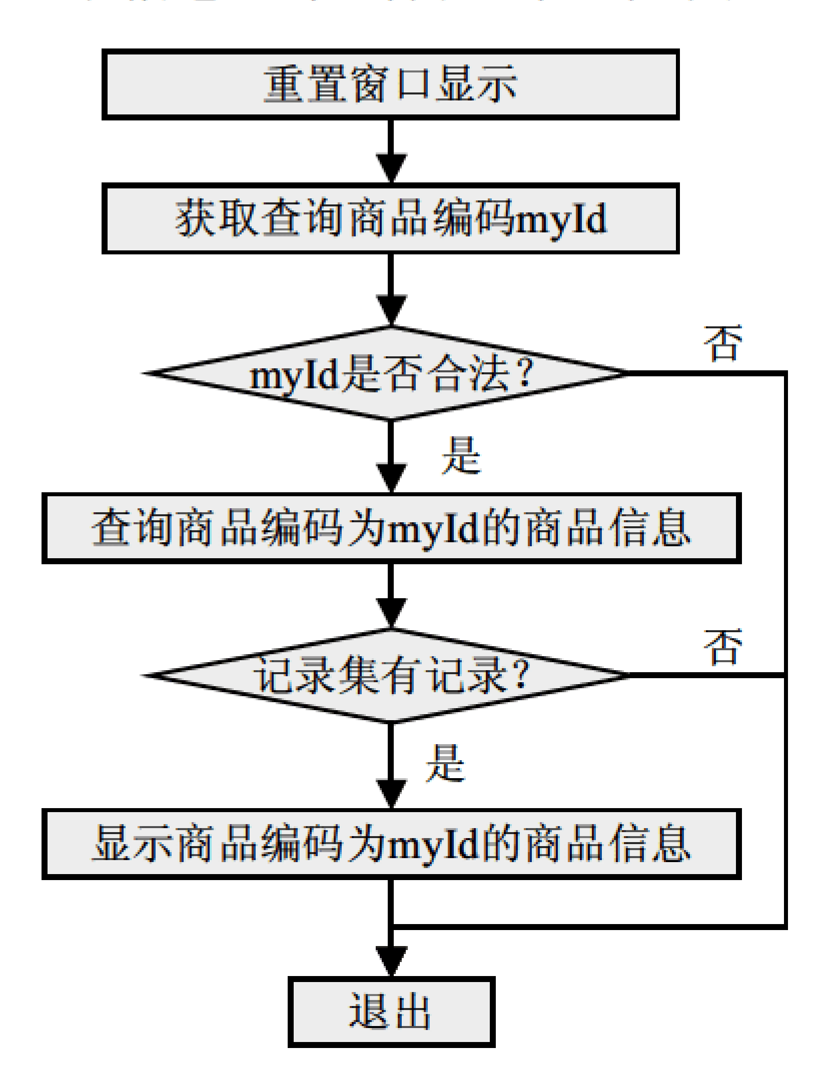


图 7-26　【查询】按钮单击事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 查询记录_Click()
    Dim myId As String
    Dim SQL As String
    Dim i As Integer
    Dim rsSerch As New ADODB.Recordset
    新建记录_Click
    myId = InputBox("请输入商品编码：", "商品查询")
    If Len(Trim(myId)) = 0 Then
        SQL = "select * from 商品信息 order by 商品编码"
    Else
```

```
        SQL = "select * from 商品信息 where 商品编码='" & myId & "'"
    End If
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If rs.BOF And rs.EOF Then
        MsgBox "没有编码为<" & myId & ">的商品信息!", vbCritical, "查询结果"
    Else
        显示商品信息
    End If
End Sub
```

7.6.8 ListView 控件项目单击事件代码设计

在 ListView 控件中单击某个项目后，程序要将该项目的所有数据显示到窗口上部对应的文本框中。程序首先将各个文本框的值清空，然后通过 For 循环，使用 ListView 控件的 ListItems 属性定位项目中的各个子项目，并把这些子项的数据显示到窗口上面对应的文本框中。以下是该过程的详细代码解释：

```
Private Sub ListView1_ItemClick(ByVal Item As MSComctlLib.ListItem)
    On Error Resume Next
    Dim i As Integer
    Call 新建记录_Click
    Me.Controls(myArray(0)).Value = ListView1.ListItems(Item.Index)
    For i = 1 To UBound(myArray)
        Me.Controls(myArray(i)).Value = ListView1.ListItems(Item.Index).SubItems(i)
    Next i
End Sub
```

7.6.9 查询与显示商品信息过程代码设计

查询与显示商品信息两个过程共同完成查询记录集并将记录集数据显示到窗口的工作。查询商品信息过程先生成一个查询商品信息字符串，然后按照该查询字符串打开记录集，从而获取对应查询信息。显示供货商信息过程使用在查询供货商信息过程中获取的供货商信息记录集，将该记录集首条记录数据写入到窗口的各个对应文本框中。以下是这两个过程的详细代码解释：

```
Public Sub 查询商品信息()
    SQL = "select * from 商品信息 order by 商品编码"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
End Sub

Public Sub 显示商品信息()
    On Error Resume Next
    Dim i As Integer
    Call 新建记录_Click
```

```
        '显示第一个商品信息
        rs.MoveFirst
        For i = 0 To UBound(myArray)
            If IsNull(rs.Fields(i)) Then
                Me.Controls(myArray(i)).Value = ""
            Else
                Me.Controls(myArray(i)).Value = rs.Fields(i)
            End If
        Next
End Sub
```

7.6.10　myListView 过程代码设计

myListView 过程完成 ListView 控件的显示设置以及标题任务，然后将记录集 rs 的所有记录数据显示到控件上。如图 7-27 所示是该过程的流程图。

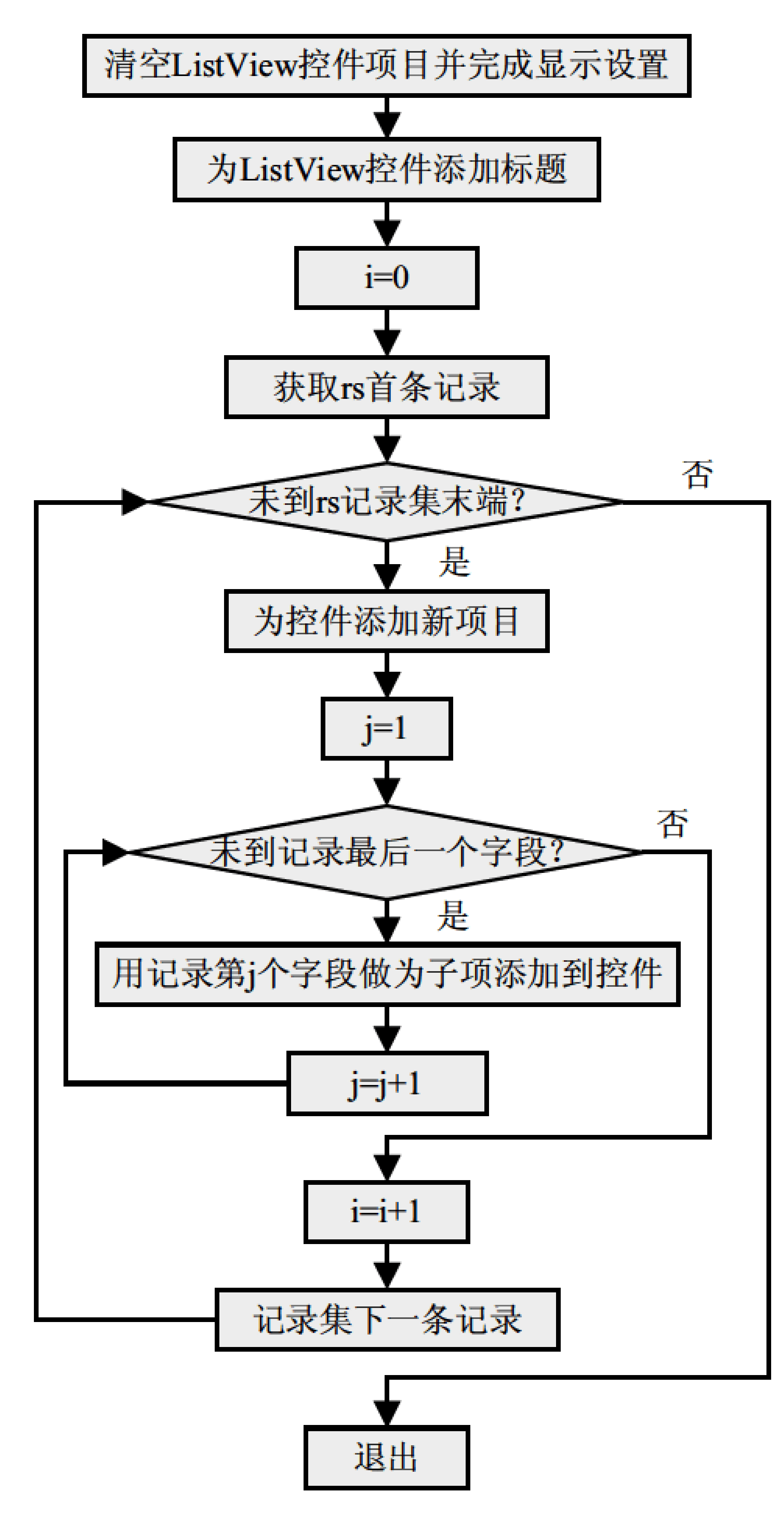


图 7-27　myListView 过程流程图

以下是该过程的详细代码解释：

```
Private Sub myListView()
    On Error Resume Next
    Dim i As Integer
    '设置 ListView 的标题
    With ListView1
        .ColumnHeaders.Clear
        .ListItems.Clear
        .View = lvwReport
        .FullRowSelect = True
        .Gridlines = True
        .Sorted = True
        .ColumnHeaders.Add , , myArray(0)
        For i = 1 To UBound(myArray)
            .ColumnHeaders.Add , , myArray(i)
        Next i
        '设置 ListView 的各行数据
        i = 0
        rs.MoveFirst
        Do While Not rs.EOF
            .ListItems.Add , , rs.Fields(0).Value
            For j = 1 To rs.Fields.Count -1
                .ListItems(i + 1).SubItems(j) = rs.Fields(j).Value
            Next j
            rs.MoveNext
            i = i + 1
        Loop
    End With
    rs.MoveFirst
    商品数目.Caption = "目前数据库中共有 " & i & " 条商品记录"
End Sub
```

7.7　进货资料管理窗体设计

【进货管理】窗口主要完成进货、进货查询和导出等操作。进货操作通过【进货资料管理】窗体实现。进货查询和导出通过【资料查询与导出】窗体实现。【资料查询与导出】窗体将会被反复使用在所有的资料查询与导出中。【资料查询与导出】窗体的相关内容将集中到查询与导出模块一起介绍。

7.7.1　进货资料管理窗体界面设计

进货所涉及到的项目包括商品名称、进货编码、商品编码、供货商编码、商品规格、计量单位、进货数量、进货单价、进货日期以及备注项目。该窗体的界面如图 7-28 所示。

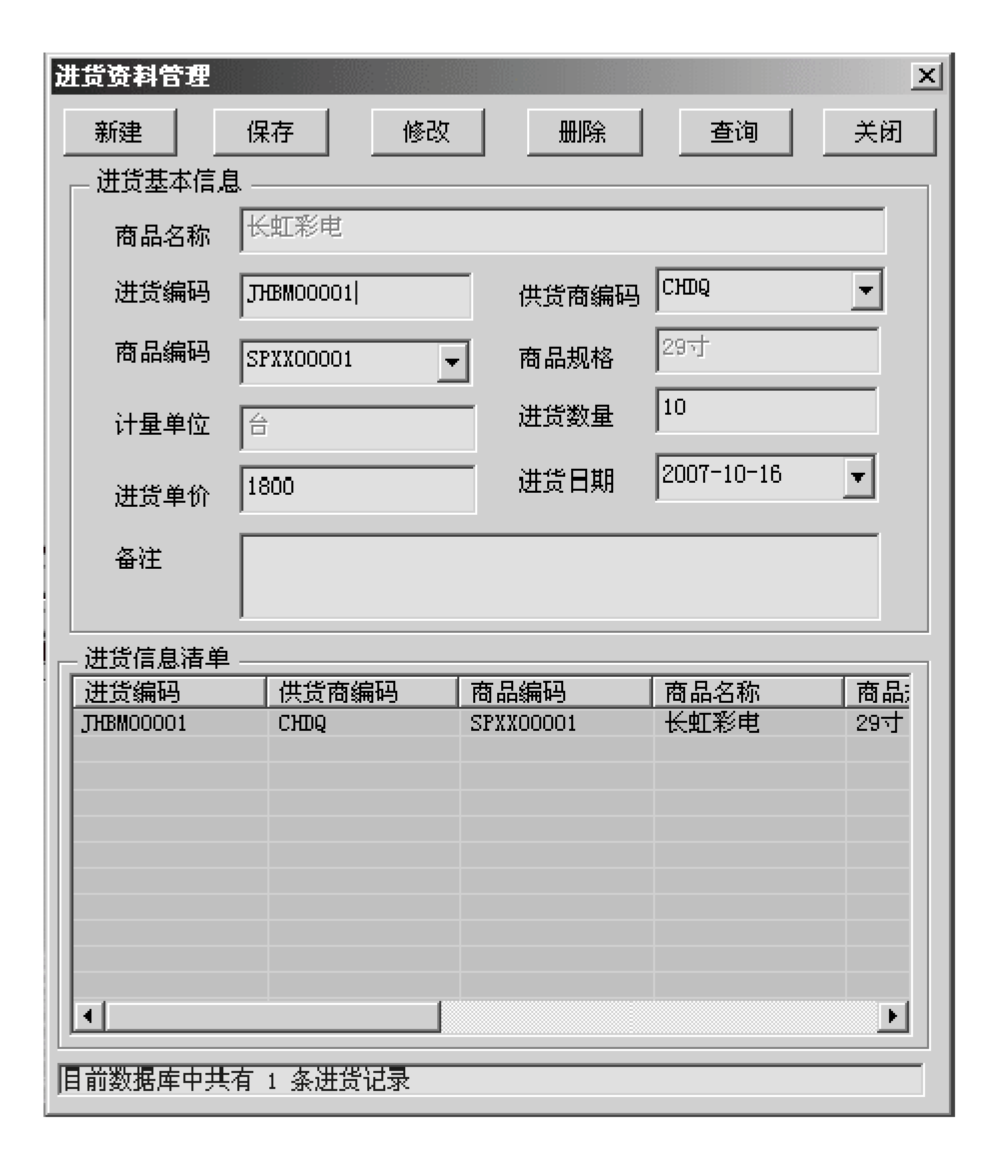


图 7-28　进货资料管理界面

7.7.2　窗口初始化与关闭事件代码设计

窗口初始化时，需要初始化窗口中使用到的数组 myArray、建立到数据库的链接、为复合框添加项目以及刷新窗口显示。刷新窗口通过 3 个过程完成：查询进货信息过程获取进货信息记录集，显示进货信息过程将第一条记录数据显示在窗口的各个输入控件中，myListView 过程将所有进货信息显示在进货信息清单中。该过程的流程图如图 7-29 所示。

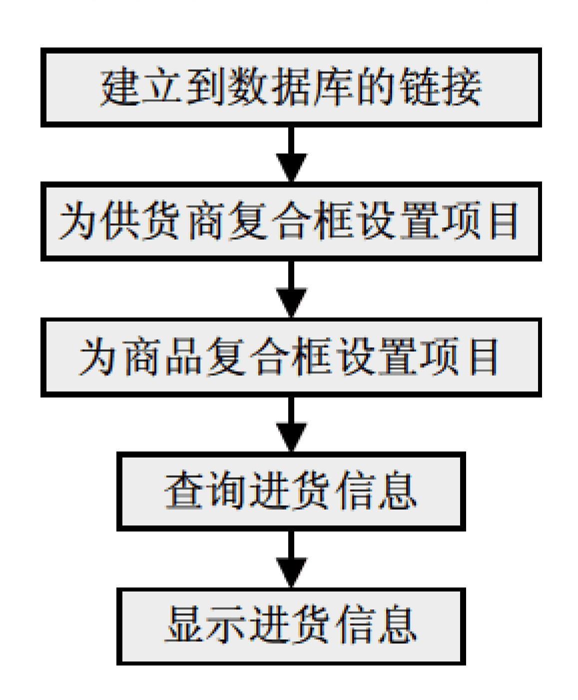


图 7-29　进货资料管理窗口初始化流程图

以下是该窗口的初始化过程与关闭事件过程详细代码解释：

```
Dim myArray As Variant
Dim cnn As New ADODB.Connection
Dim rs As ADODB.Recordset

Private Sub UserForm_Initialize()
    Dim i As Integer
    Dim SQL As String
    Dim rsx As ADODB.Recordset
```

```
    myArray = Array("进货编码", "供货商编码", "商品编码", "商品名称", _
        "商品规格", "计量单位", "进货数量", "进货单价", "进货日期", "备注")
    进货日期.Value = Date
    '建立与数据库的连接
    With cnn
        .ConnectionString = "Provider=microsoft.jet.oledb.4.0;" _
        & "Data Source=" & ThisWorkbook.Path & "\进销存数据库.mdb;" _
        & "Jet Oledb:database password=123456;"
        .Open
    End With
    '为供货商编码复合框设置项目
    SQL = "select 供货商编码 from 供货商信息 order by 供货商编码"
    Set rsx = New ADODB.Recordset
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    With 供货商编码
        .Clear
        Do While Not rsx.EOF
            .AddItem rsx!供货商编码
            rsx.MoveNext
        Loop
    End With
    On Error Resume Next
    供货商编码.ListIndex = 0
    On Error GoTo 0
    '为商品编码复合框设置项目
    SQL = "select 商品编码 from 商品信息 order by 商品编码"
    Set rsx = New ADODB.Recordset
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    With 商品编码
        .Clear
        Do While Not rsx.EOF
            .AddItem rsx!商品编码
            rsx.MoveNext
        Loop
    End With
    On Error Resume Next
    商品编码.ListIndex = 0
    On Error GoTo 0
    '查询并在窗体上显示数据
    查询进货信息
    显示进货信息
    myListView
End Sub
```

7.7.3　保存按钮单击事件代码设计

【保存】按钮用于将用户新建的进货资料保存到数据库中。该按钮单击事件过程首先检

测用户是否输入了必要的进货数据，然后确认进货编号与数据库已有编号是否重复，随后程序还需要确认各个数据的长度不超过数据库允许长度，最后程序将商品资料保存到数据库中并刷新窗口显示。如图 7-30 所示的是该过程的流程图。

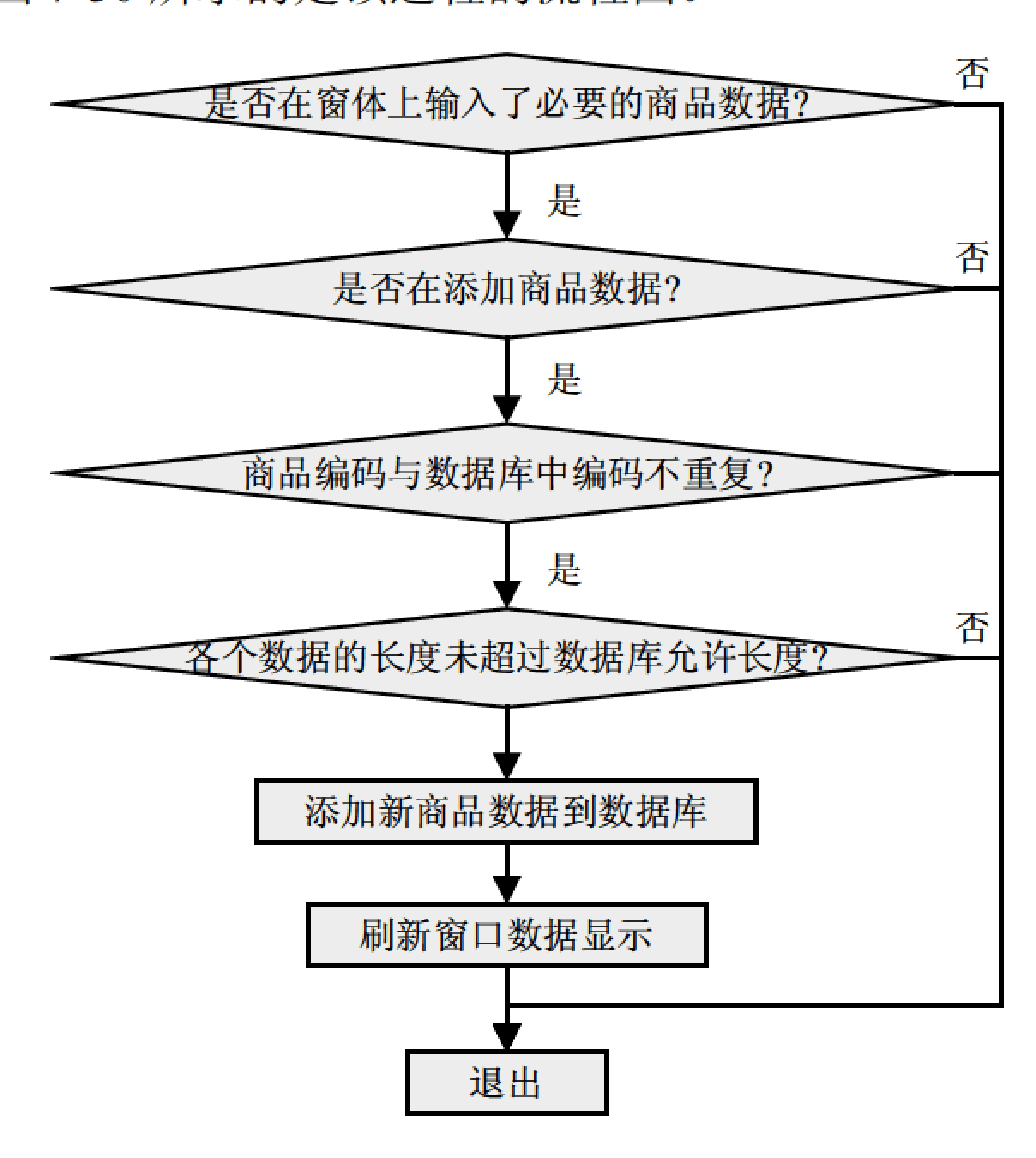


图 7-30　【保存】按钮单击事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 保存记录_Click()
    Dim i As Integer
    '判断是否在窗体上输入了必要的进货数据
    For i = 0 To UBound(myArray) -1
        If Me.Controls(myArray(i)).Name <> "备注" Then
            If Me.Controls(myArray(i)).Value = "" Then
                MsgBox Me.Controls(myArray(i)).Name & "不能为空！", vbCritical
                Me.Controls(myArray(i)).SetFocus
                Exit Sub
            End If
        End If
    Next i
    If MsgBox("本操作将添加新的进货记录！" & vbCrLf & "是否要添加？", _
        vbQuestion + vbYesNo, "添加记录") = vbNo Then Exit Sub
    '首先判断在数据库中是否存在相同的进货编码
    Dim rsNum As New ADODB.Recordset
    SQL = "select 进货编码 from 进货信息 where 进货编码='" & 进货编码.Value & "'"
    rsNum.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If rsNum.BOF = False And rsNum.EOF = False Then
        MsgBox "在数据库中已经存在有编号为<" & 进货编码.Value & ">的进货记录！" _
```

```
            & vbCrLf & "请重新输入进货编码！ ", vbOKOnly + vbCritical
        Me.进货编码.Value = ""
        Me.进货编码.SetFocus
        GoTo hhh
    End If
    '----准备将窗体上的数据添加到数据库中----
    SQL = "select * from 进货信息"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    '判断各个数据的长度是否超过了数据库允许的长度
    For i = 0 To UBound(myArray)
        If Len(Me.Controls(myArray(i)).Value) > rs.Fields(i).DefinedSize Then
            MsgBox Me.Controls(myArray(i)).Name _
                & "的数据长度已经超过了数据库规定的长度！ ", vbCritical
            Me.Controls(myArray(i)).Value = Left(Me.Controls(myArray(i)).Value, _
                rs.Fields(i).DefinedSize)
            Me.Controls(myArray(i)).SetFocus
            Exit Sub
        End If
    Next i
    '开始添加数据
    With rs
        .AddNew
        For i = 0 To UBound(myArray)
            .Fields(i) = Me.Controls(myArray(i)).Value
        Next i
        .Update
    End With
    MsgBox "已经成功将新进货数据添加到数据库中！ ", vbInformation, "添加记录"
    '刷新查询和显示
    查询进货信息
    显示进货信息
    myListView
hhh:
    rsNum.Close
    Set rsNum = Nothing
End Sub
```

7.7.4　进货数量文本框事件代码设计

对于每一个商品都存在一个最大与最小库存量。在进货数量文本框中输入进货数量时，程序需要统计剩余库存，将该数据加上进货数获取进货后库存数。如果进货后库存数仍然小于最小库存，用户需要加大进货量。如果进货后库存数超过最大库存，用户需要减少进货数。这些功能都是通过进货数量文本框事件实现的。如图 7-31 所示的是该事件过程的流程图。

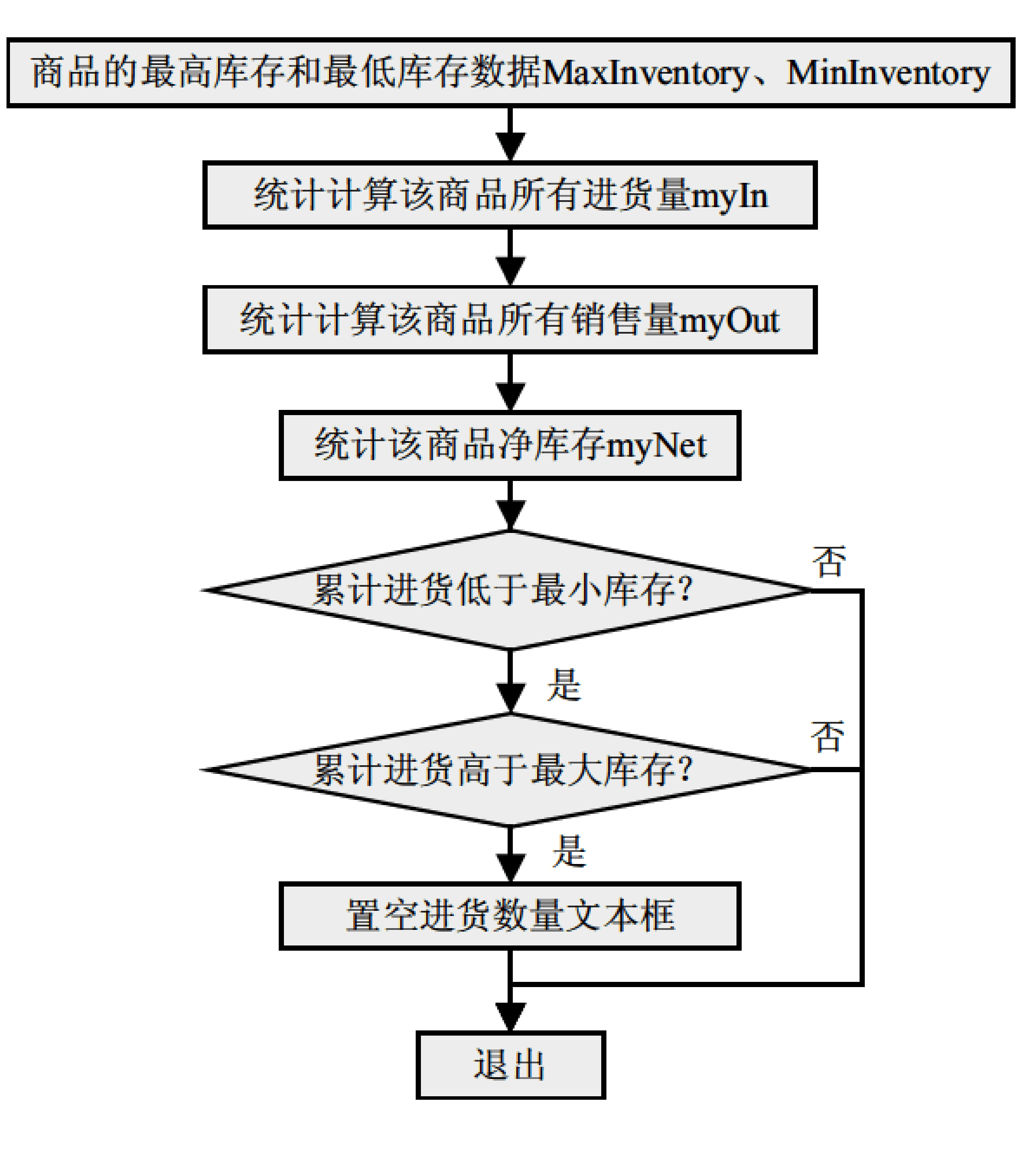


图 7-31 进货数量文本框事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 进货数量_Exit(ByVal Cancel As MSForms.ReturnBoolean)
    On Error Resume Next
    Dim rsx As ADODB.Recordset
    Dim SQL As String
    Dim MaxInventory As Integer, MinInventory As Integer
    Dim myOut As Integer, myIn As Integer, myNet As Integer
    '查询该商品的最高库存和最低库存数据
    SQL = "select 最高库存,最低库存 from 商品信息 " _
        & "where 商品编码='" & 商品编码.Value & "'"
    Set rsx = New ADODB.Recordset
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If IsNull(rsx!最高库存) Then
        MaxInventory = 0
    Else
        MaxInventory = rsx!最高库存
    End If
    If IsNull(rsx!最低库存) Then
        MinInventory = 0
    Else
        MinInventory = rsx!最低库存
    End If
    '统计计算该商品目前的库存
    SQL = "select sum(进货数量) as aa from 进货信息 where " _
        & "商品编码='" & 商品编码.Value & "'"
    Set rsx = New ADODB.Recordset
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic
```

```
    If IsNull(rsx!aa) Then
        myIn = 0
    Else
        myIn = rsx!aa
    End If
    SQL = "select sum(销售数量) as aa from 销售信息 " _
        & "where 商品编码='" & 商品编码.Value & "'"
    Set rsx = New ADODB.Recordset
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If IsNull(rsx!aa) Then
        myOut = 0
    Else
        myOut = rsx!aa
    End If
    myNet = myIn -myOut
    If Val(进货数量.Value) + myNet > MaxInventory Then
        MsgBox "当前该商品库存为 " & myNet & "！" _
            & vbCrLf & "累计进货数量已经超过了最高库存！" _
            & vbCrLf & "请重新输入进货数量", vbCritical, "进货数量"
        进货数量.Value = ""
        进货数量.SetFocus
        Exit Sub
    ElseIf Val(进货数量.Value) + myNet < MinInventory Then
        MsgBox "当前该商品库存为 " & myNet & "！" _
            & vbCrLf & "累计进货数量不足！" _
            & vbCrLf & "请重新输入进货数量", vbCritical, "进货数量"
        进货数量.Value = ""
        进货数量.SetFocus
        Exit Sub
    End If
End Sub
```

7.7.5　商品编码复合框事件代码设计

当用户在窗口的商品编码复合框中选定了某个项目后，程序将会刷新窗口中有关该商品信息的输入框（其中包括商品名称、商品规格和计量单位）。为了保证这些控件的数据与商品编码同步，程序不允许用户编辑这些控件中的数据。以下是该过程的详细代码解释：

```
Private Sub 商品编码_Change()
    On Error Resume Next
    Dim rsx As New ADODB.Recordset
    '为商品名称复合框设置项目
    SQL = "select * from 商品信息 where 商品编码='" & 商品编码.Value & "'"
    Set rsx = New ADODB.Recordset
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    商品名称 = rsx!商品名称
    商品规格 = rsx!商品规格
    计量单位 = rsx!计量单位
```

```
        商品名称.Enabled = False
        商品规格.Enabled = False
        计量单位.Enabled = False
        rsx.Close
        Set rsx = Nothing
End Sub
```

7.7.6　新建按钮单击事件代码设计

【新建】按钮用于重设窗口中所有输入框，便于用户输入新进货信息。窗口中所有的输入控件的名称在窗口初始化过程中已经保存，程序通过一个 For 循环遍历窗口所有输入控件。以下是该过程的详细代码解释：

```
Private Sub 新建记录_Click()
    For i = 0 To UBound(myArray)
        If Me.Controls(myArray(i)).Name <> "进货日期" Then
            Me.Controls(myArray(i)).Value = ""
        Else
            Me.Controls(myArray(i)).Value = Date
        End If
    Next i
    进货日期 = Date
    进货编码.Enabled = True
    进货编码.SetFocus
End Sub
```

7.7.7　修改按钮单击事件代码设计

【修改】按钮将用户对商品资料做出的修改操作保存到数据库中。在确认用户需要修改数据后，程序通过一个更新查询 SQL 语句完成修改数据库记录操作，然后通过调用查询商品信息、显示商品信息与 myListView 过程刷新窗口的数据显示。如图 7-32 所示是该过程的流程图。

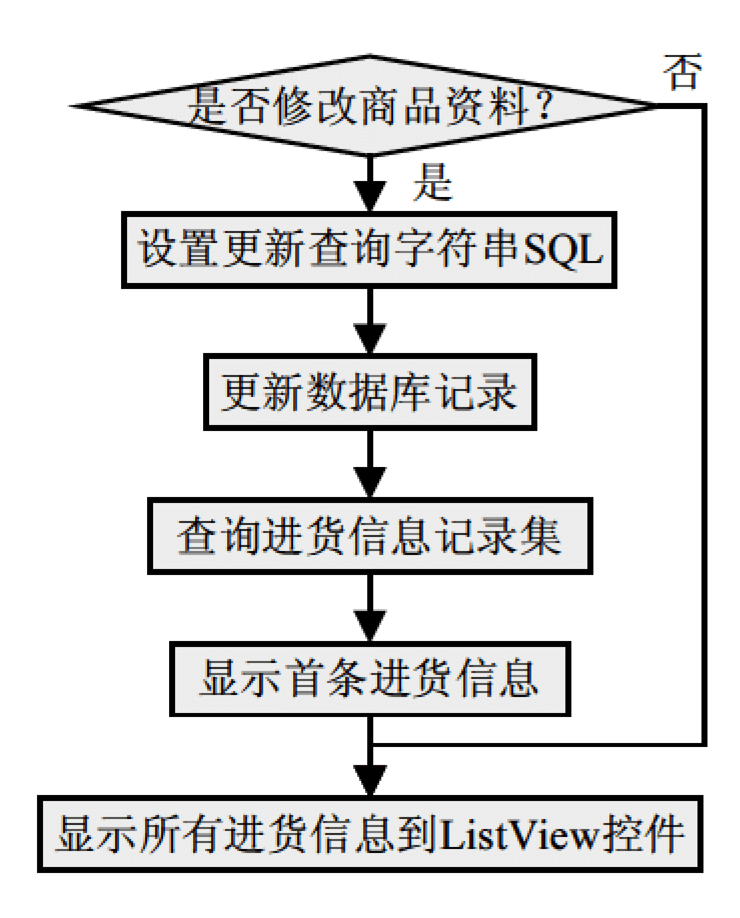


图 7-32　【修改】按钮单击事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 修改记录_Click()
    If MsgBox("本操作将修改编码为<" & 进货编码.Value & ">的进货信息记录！" _
        & vbCrLf & "是否要修改？", _
        vbQuestion + vbYesNo + vbDefaultButton2, "修改记录") = vbNo Then Exit Sub
    '----修改更新进货信息记录----
    SQL = "update 进货信息 set " _
        & "商品编码='" & 商品编码.Value & "'," _
        & "商品编码='" & 商品编码.Value & "'," _
        & "商品名称='" & 商品名称.Value & "'," _
        & "商品规格='" & 商品规格.Value & "'," _
        & "计量单位='" & 计量单位.Value & "'," _
        & "进货数量='" & 进货数量.Value & "," _
        & "进货单价='" & 进货单价.Value & "," _
        & "进货日期='" & 进货日期.Value & "'," _
        & "备注='" & 备注.Value & "' " _
        & "where 进货编码='" & 进货编码.Value & "'"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    MsgBox "已经成功将编码为<" & 进货编码.Value & ">的进货信息记录进行修改！", _
        vbInformation, "修改记录"
    '刷新查询和显示
    查询进货信息
    显示进货信息
    myListView
End Sub
```

7.7.8 删除按钮单击事件代码设计

【删除】按钮用于删除当前显示的商品资料记录，程序通过一个删除查询在数据库中完成删除操作。在删除操作完成之后，程序刷新了窗口显示的记录数据。以下是该过程的详细代码解释：

```
Private Sub 删除记录_Click()
    On Error Resume Next
    If MsgBox("本操作将删除编码为<" & 进货编码.Value & ">的进货信息记录！" _
        & vbCrLf & "是否要删除？", _
        vbQuestion + vbYesNo + vbDefaultButton2, "删除记录") = vbNo Then Exit Sub
    SQL = "delete from 进货信息 where 进货编码='" & 进货编码.Value & "'"
    Set rs = cnn.Execute(SQL)
    MsgBox "已经成功将编码为<" & 进货编码.Value & ">的进货信息记录删除！", _
        vbInformation, "删除记录"
    '刷新查询和显示
    Call 查询进货信息
    Call 显示进货信息
    Call myListView
End Sub
```

7.7.9　查询按钮单击事件代码设计

在进货信息的查询操作中只能按照进货编码进行查询。单击【查询】按钮后，弹出一个进货编码输入框。用户输入完进货编码后，程序从数据库中获取进货编码为用户输入结果的所有进货信息，并将查询到的记录信息显示到窗口中。如图 7-33 所示的是该过程的流程图。

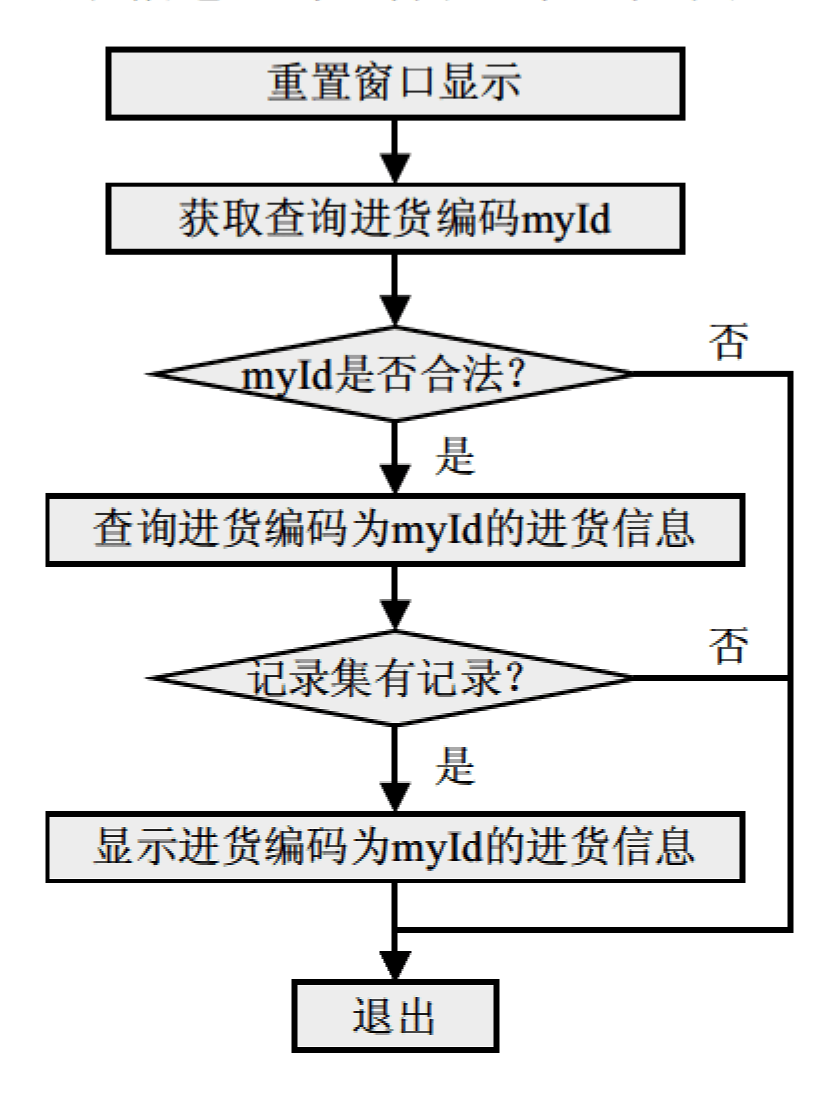


图 7-33　【查询】按钮单击事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 查询记录_Click()
    Dim myId As String
    Dim SQL As String
    Dim i As Integer
    Dim rsSerch As New ADODB.Recordset
    Call 新建记录_Click
    myId = InputBox("请输入进货编码：", "进货查询")
    If Len(Trim(myId)) = 0 Then
        SQL = "select * from 进货信息 order by 进货编码"
    End If
    SQL = "select * from 进货信息 where 进货编码='" & myId & "'"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If rs.BOF And rs.EOF Then
        MsgBox "没有编码为<" & myId & ">的进货信息!", vbCritical, "查询结果"
    Else
        Call 显示进货信息
        进货编码.Enabled = False
    End If
End Sub
```

7.7.10 ListView 控件项目单击事件代码设计

用户在 ListView 控件中单击某个项目后，程序要将该项目的所有数据显示到窗口上部对应的文本框中。程序首先将各个文本框的值清空，然后通过 For 循环，使用 ListView 控件的 ListItems 属性定位项目里各个子项目，并把这些子项目的数据显示到窗口上面对应的文本框中。以下是该过程的详细代码解释：

```
Private Sub ListView1_ItemClick(ByVal Item As MSComctlLib.ListItem)
    On Error Resume Next
    Dim i As Integer
    Call 新建记录_Click
    进货编码.Enabled = False
    Me.Controls(myArray(0)).Value = ListView1.ListItems(Item.Index)
    For i = 1 To UBound(myArray)
        Me.Controls(myArray(i)).Value = ListView1.ListItems(Item.Index).SubItems(i)
    Next i
End Sub
```

7.7.11 查询与显示进货信息过程代码设计

查询与显示进货信息两个过程共同完成查询记录集并将记录集数据显示到窗口的工作。查询进货信息首先生成一个查询进货信息字符串，然后按照该查询字符串打开记录集，从而获取对应查询信息。显示进货信息过程使用在查询进货信息过程中获取的供货商信息记录集，将该记录集首条记录数据写入到窗口的各个对应文本框中。以下是这两个过程的详细代码解释：

```
Public Sub 查询进货信息()
    SQL = "select * from 进货信息 order by 进货编码"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
End Sub

Public Sub 显示进货信息()
    On Error Resume Next
    Dim i As Integer
    Call 新建记录_Click
    rs.MoveFirst
    For i = 0 To UBound(myArray)
        If IsNull(rs.Fields(i)) Then
            Me.Controls(myArray(i)).Value = ""
        Else
            Me.Controls(myArray(i)).Value = rs.Fields(i)
        End If
    Next i
End Sub
```

7.7.12　myListView 过程代码设计

myListView 过程完成 ListView 控件的显示设置以及标题显示任务，然后程序将记录集 rs 的所有记录数据显示到控件上。如图 7-34 所示是该过程的流程图。

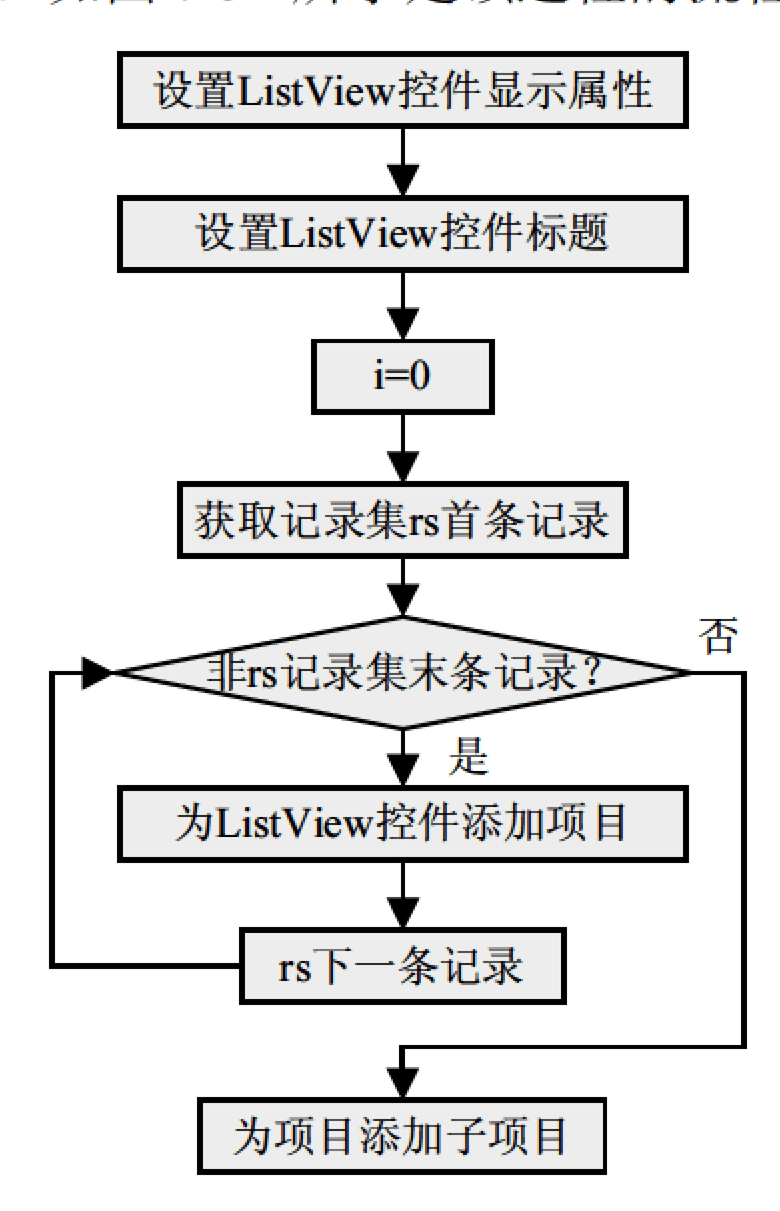


图 7-34　myListView 过程流程图

以下是该过程的详细代码解释：

```
Private Sub myListView()
    On Error Resume Next
    Dim i As Integer
    '设置 ListView 的标题
    With ListView1
        .ColumnHeaders.Clear
        .ListItems.Clear
        .View = lvwReport
        .FullRowSelect = True
        .Gridlines = True
        .Sorted = True
        .ColumnHeaders.Add , , myArray(0)
        For i = 1 To UBound(myArray)
            .ColumnHeaders.Add , , myArray(i)
        Next
        '设置 ListView 的各行数据
        i = 0
        rs.MoveFirst
        Do While Not rs.EOF
            .ListItems.Add , , rs.Fields(0).Value
            For j = 1 To rs.Fields.Count -1
```

```
                .ListItems(i + 1).SubItems(j) = rs.Fields(j).Value
            Next
            rs.MoveNext
            i = i + 1
        Loop
    End With
    rs.MoveFirst
    进货记录.Caption = "目前数据库中共有 " & i & " 条进货记录"
End Sub
```

7.8 销售资料管理窗体设计

销售管理窗口主要完成销售、销售查询和导出。两个功能都是通过独立的窗体来实现的：销售工作通过销售资料管理窗体实现，销售查询和导出通过资料查询与导出窗体实现。资料查询与导出窗体将会被反复使用在所有的资料查询与导出中，其相关介绍将集中到查询与导出模块一起介绍。

7.8.1 销售资料管理窗体界面设计

销售所涉及到的项目包括商品名称、销售编码、商品编码、商品规格、计量单位、销售数量、销售单价、销售日期以及备注项目。该窗体的界面如图 7-35 所示。

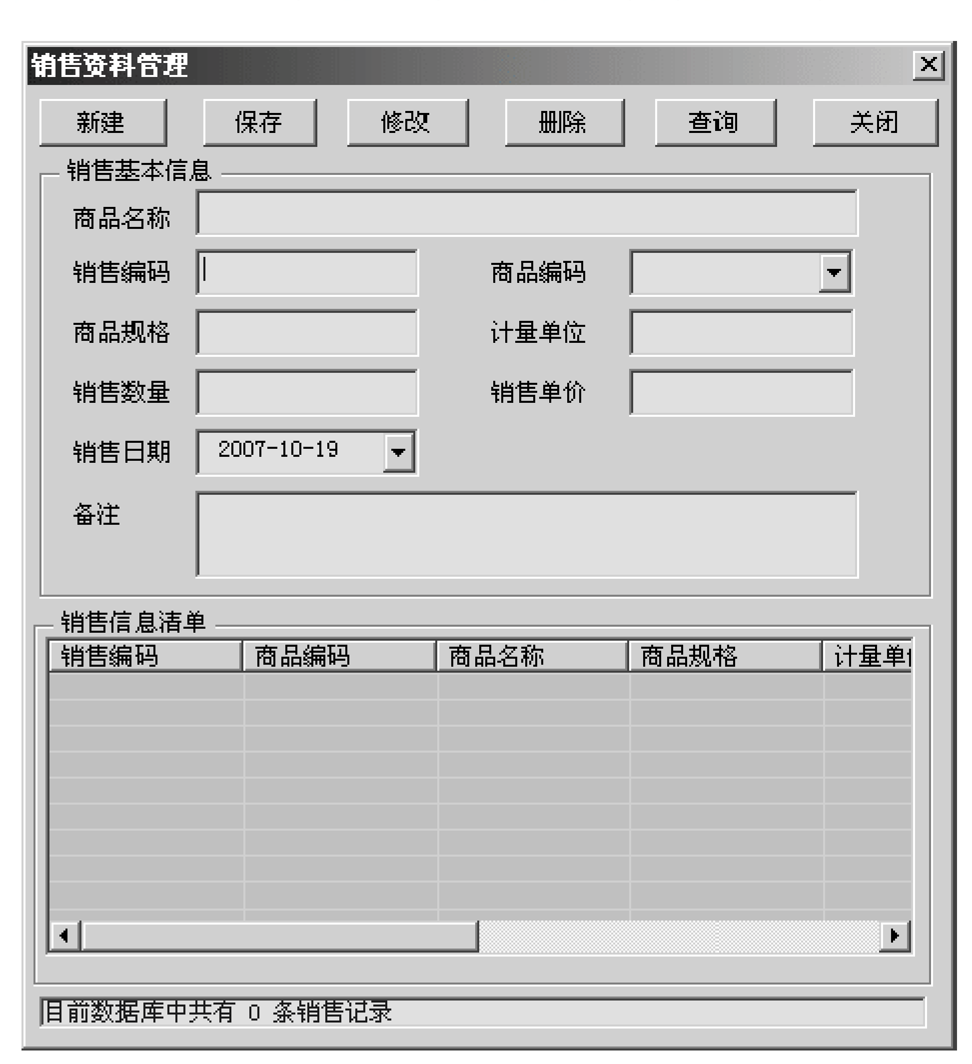


图 7-35 销售资料管理界面

7.8.2 窗口初始化与关闭事件代码设计

窗口初始化时需要初始化窗口中使用到的数组 myArray、建立到数据库的链接、为复合框

添加项目以及刷新窗口显示。刷新窗口通过 3 个过程完成：查询销售信息过程获取销售信息记录集，显示销售信息过程将第一条记录数据显示在窗口的各个输入控件中，myListView 过程将所有销售信息显示在销售信息清单中。该过程的流程图如图 7-36 所示。

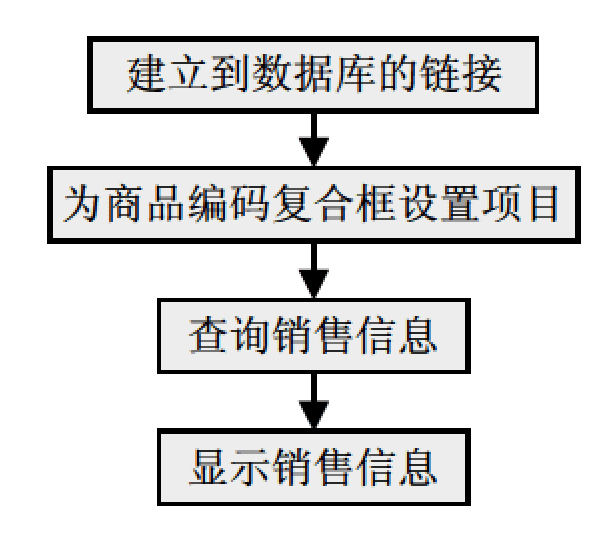


图 7-36　进货资料管理窗口初始化流程图

以下是该窗体的初始化过程与关闭事件过程的详细代码解释：

```
Dim myArray As Variant
Dim cnn As New ADODB.Connection
Dim rs As ADODB.Recordset

Private Sub UserForm_Initialize()
    Dim i As Integer
    Dim SQL As String
    Dim rsx As ADODB.Recordset
    myArray = Array("销售编码", "商品编码", "商品名称", "商品规格", _
        "计量单位", "销售数量", "销售单价", "销售日期", "备注")
    销售日期.Value = Date
    '建立与数据库的连接
    With cnn
        .ConnectionString = "Provider=microsoft.jet.oledb.4.0;" _
        & "Data Source=" & ThisWorkbook.Path & "\进销存数据库.mdb;" _
        & "Jet Oledb:database password=123456;"
        .Open
    End With
    '为商品编码复合框设置项目
    SQL = "select 商品编码 from 商品信息 order by 商品编码"
    Set rsx = New ADODB.Recordset
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    With 商品编码
        .Clear
        Do While Not rsx.EOF
            .AddItem rsx!商品编码
            rsx.MoveNext
        Loop
    End With
    rsx.Close
    Set rsx = Nothing
    '查询并在窗体上显示数据
    查询销售信息
    显示销售信息
    myListView
End Sub

Private Sub 关闭退出_Click()
    cnn.Close
```

```
        Set rs = Nothing
        Set cnn = Nothing
        Unload 销售资料管理
End Sub
```

7.8.3 保存按钮单击事件代码设计

【保存】按钮用于将用户新建的销售商品资料保存到数据库中。该按钮单击事件过程首先检测用户是否输入了必要的销售数据，然后确认销售编号与数据库已有编号是否重复，随后程序还需要确认各个数据的长度不超过数据库允许长度，最后程序将销售资料保存到数据库中并刷新窗口显示。如图 7-37 所示的是该过程的流程图。

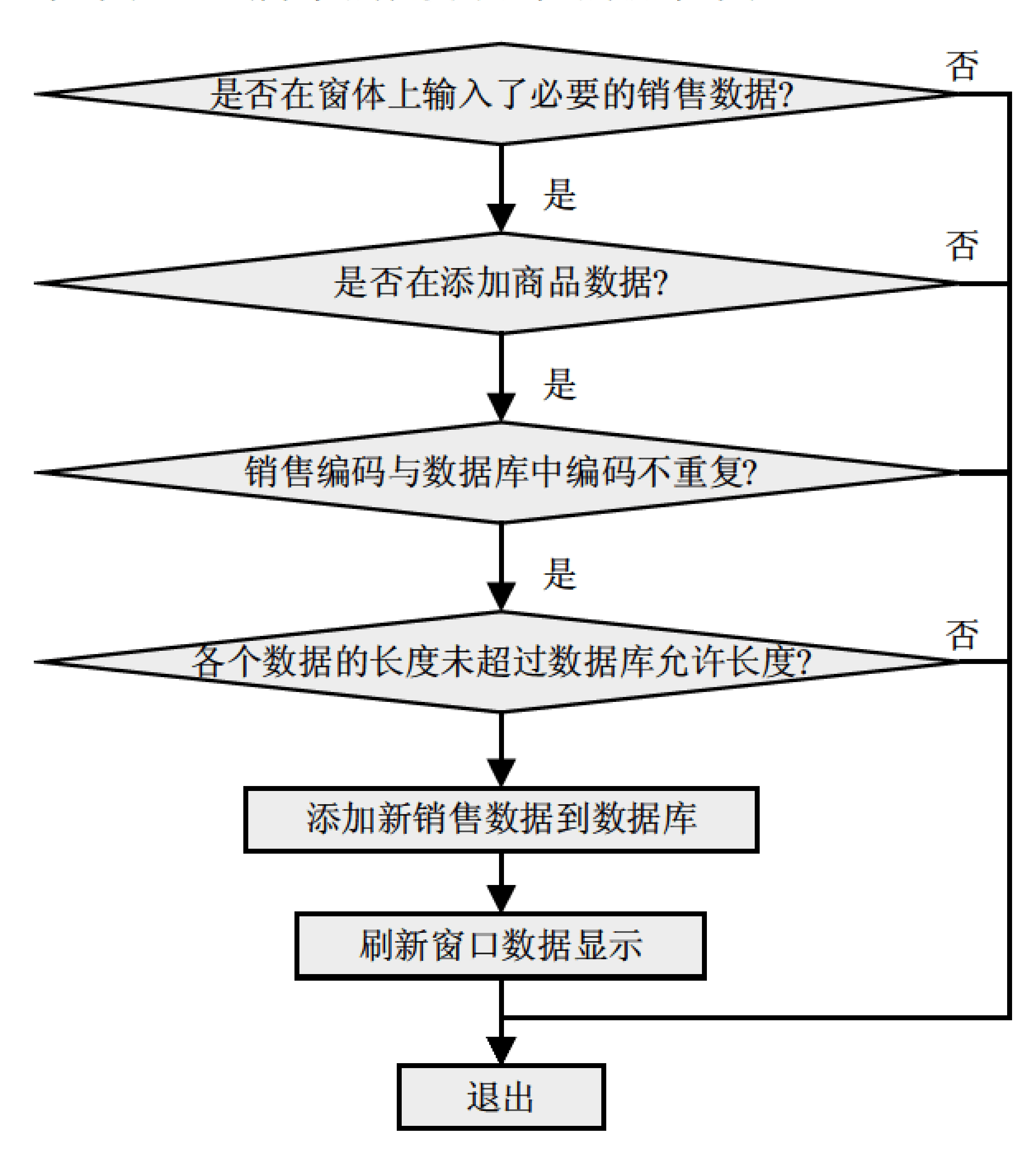


图 7-37 【保存】按钮单击事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 保存记录_Click()
        Dim i As Integer
        '判断是否在窗体上输入了必要的销售数据
        For i = 0 To UBound(myArray) -1
                If Me.Controls(myArray(i)).Name <> "备注" Then
                        If Me.Controls(myArray(i)).Value = "" Then
                                MsgBox Me.Controls(myArray(i)).Name & "不能为空！", vbCritical
                                Me.Controls(myArray(i)).SetFocus
                                Exit Sub
                        End If
                End If
```

```
    Next
    If MsgBox("本操作将添加新的销售记录！" & vbCrLf & "是否要添加？", _
        vbQuestion + vbYesNo, "添加记录") = vbNo Then Exit Sub
    '首先判断在数据库中是否存在相同的销售编码
    Dim rsNum As New ADODB.Recordset
    SQL = "select 销售编码 from 销售信息 where 销售编码='" & 销售编码.Value & "'"
    rsNum.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If rsNum.BOF = False And rsNum.EOF = False Then
        MsgBox "在数据库中已经存在有编号为<" & 销售编码.Value & ">的销售记录！" _
            & vbCrLf & "请重新输入销售编码！", vbOKOnly + vbCritical
        Me.销售编码.Value = ""
        Me.销售编码.SetFocus
        GoTo hhh
    End If
    '准备将窗体上的数据添加到数据库中
    SQL = "select * from 销售信息"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    '判断各个数据的长度是否超过了数据库允许的长度
    For i = 0 To UBound(myArray)
        If Len(Me.Controls(myArray(i)).Value) > rs.Fields(i).DefinedSize Then
            MsgBox Me.Controls(myArray(i)).Name _
                & "的数据长度已经超过了数据库规定的长度！", vbCritical
            Me.Controls(myArray(i)).Value = Left(Me.Controls(myArray(i)).Value, _
                rs.Fields(i).DefinedSize)
            Me.Controls(myArray(i)).SetFocus
            Exit Sub
        End If
    Next i
    '开始添加数据
    With rs
        .AddNew
        For i = 0 To UBound(myArray)
            .Fields(i) = Me.Controls(myArray(i)).Value
        Next i
        .Update
    End With
    MsgBox "已经成功将新销售数据添加到数据库中！", vbInformation, "添加记录"
    '刷新查询和显示
    查询销售信息
    显示销售信息
    myListView
hhh:
    rsNum.Close
    Set rsNum = Nothing
End Sub
```

7.8.4 商品编码复合框事件代码设计

当用户在窗口的商品编码复合框中选定了某个项目后，程序将会刷新窗口中有关该商品信息的输入框。这些输入框包括商品名称、商品规格和计量单位。为了保证这些控件的数据与商品编码同步，程序不允许用户编辑这些控件中的数据。以下是该过程的详细代码解释：

```
Private Sub 商品编码_Change()
    On Error Resume Next
    Dim rsx As New ADODB.Recordset
    '为商品名称复合框设置项目
    SQL = "select * from 商品信息 where 商品编码='" & 商品编码.Value & "'"
    Set rsx = New ADODB.Recordset
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    商品名称 = rsx!商品名称
    商品规格 = rsx!商品规格
    计量单位 = rsx!计量单位
    商品名称.Enabled = False
    商品规格.Enabled = False
    计量单位.Enabled = False
    rsx.Close
    Set rsx = Nothing
End Sub
```

7.8.5 销售数量文本框事件代码设计

在用户确认销售数据时，程序需要根据库存数据判定该次销售操作是否实现。程序首先从数据库中获取该商品的总进货量，然后获取该商品的总销售量，根据这两个数据获取实际商品库存。注意，用户输入的销售商品数量不能超过该实际商品库存。如图 7-38 所示是该过程的流程图。

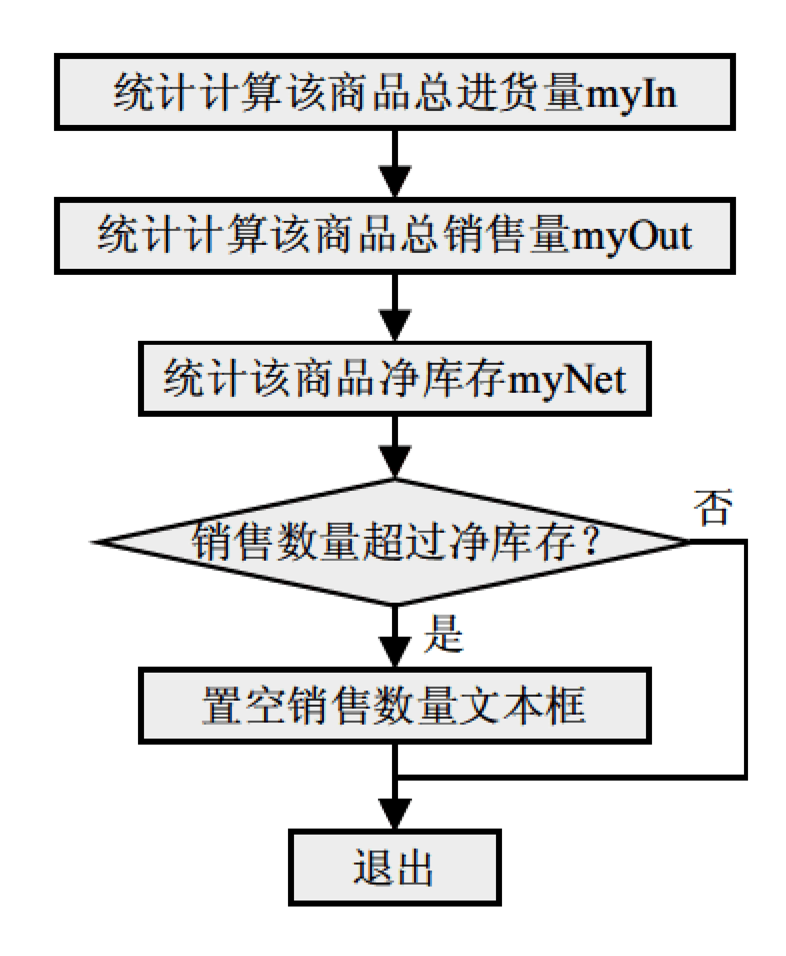


图 7-38 销售数量文本框事件代码设计

以下是该过程的详细代码解释：

```
Private Sub 销售数量_Exit(ByVal Cancel As MSForms.ReturnBoolean)
    Dim rsx As ADODB.Recordset
    Dim SQL As String
    Dim myOut As Integer, myIn As Integer, myNet As Integer
    '统计计算该商品目前的库存
    SQL = "select sum(进货数量) as aa from 进货信息 where 商品编码='" & 商品编码.Value & "'"
    Set rsx = New ADODB.Recordset
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If IsNull(rsx!aa) Then
        myIn = 0
    Else
        myIn = rsx!aa
    End If
    SQL = "select sum(销售数量) as aa from 销售信息 where 商品编码='" & 商品编码.Value & "'"
    Set rsx = New ADODB.Recordset
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If IsNull(rsx!aa) Then
        myOut = 0
    Else
        myOut = rsx!aa
    End If
    myNet = myIn -myOut
    If Val(销售数量.Value) > myNet Then
        MsgBox "当前该商品库存为 " & myNet & "！" _
            & vbCrLf & "销售数量已经超过净库存量！" _
            & vbCrLf & "请重新输入销售数量", vbCritical, "销售数量"
        销售数量.Value = ""
        销售数量.SetFocus
        Exit Sub
    End If
End Sub
```

7.8.6　新建按钮单击事件代码设计

【新建】按钮用于重设窗口中的所有输入框，便于用户输入新销售信息。窗口中所有的输入控件名称在窗口初始化过程中已经保存，程序通过一个 For 循环遍历窗口所有输入控件。除了销售日期文本框以外，其他所有输入控件都需要重置。以下是该按钮的详细代码解释：

```
Private Sub 新建记录_Click()
    For i = 0 To UBound(myArray)
        If Me.Controls(myArray(i)).Name <> "销售日期" Then
            Me.Controls(myArray(i)).Value = ""
        End If
    Next
    销售日期 = Date
    销售编码.Enabled = True
    销售编码.SetFocus
End Sub
```

7.8.7　修改按钮单击事件代码设计

【修改】按钮将用户对销售数据做出的修改操作保存到数据库中。在确认用户需要修改数据后，程序通过一个更新查询 SQL 语句完成修改数据库记录操作，通过调用查询销售信息、显示销售信息与 myListView 过程刷新窗口的数据显示。如图 7-39 所示是该过程的流程图。

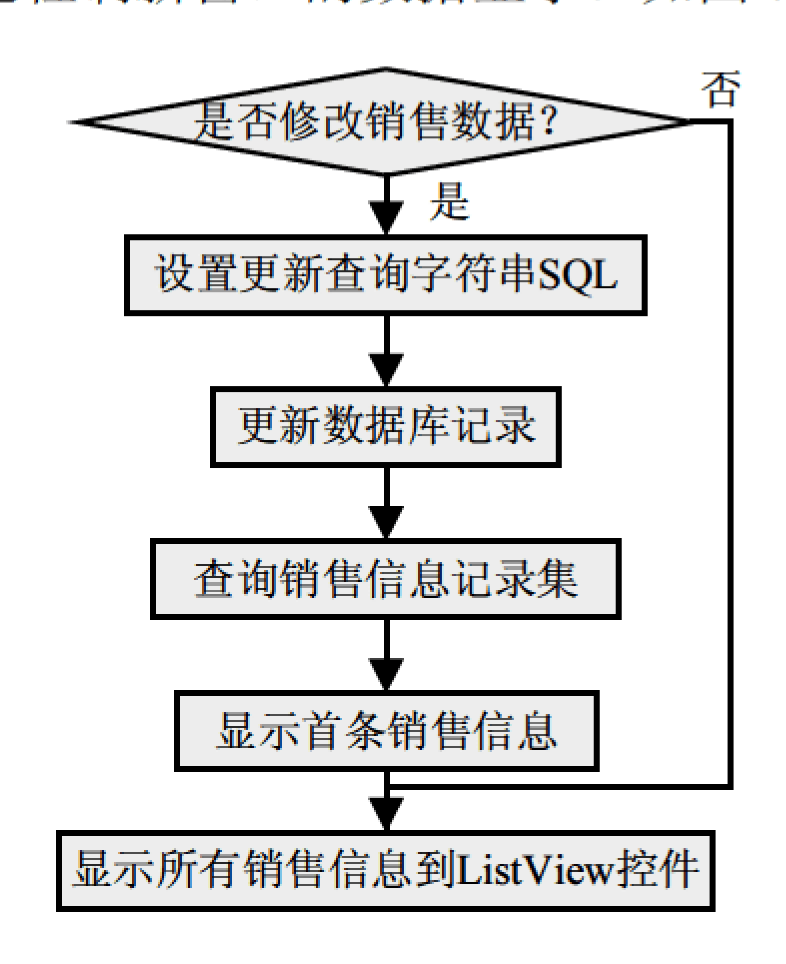


图 7-39　【修改】按钮单击事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 修改记录_Click()
    If MsgBox("本操作将修改编码为<" & 销售编码.Value & ">的销售信息记录！" _
        & vbCrLf & "是否要修改？", _
        vbQuestion + vbYesNo + vbDefaultButton2, "修改记录") = vbNo Then Exit Sub
    '修改更新记录
    SQL = "update 销售信息 set " _
        & "商品编码='" & 商品编码.Value & "'," _
        & "商品名称='" & 商品名称.Value & "'," _
        & "商品规格='" & 商品规格.Value & "'," _
        & "计量单位='" & 计量单位.Value & "'," _
        & "销售数量=" & 销售数量.Value & "," _
        & "销售单价=" & 销售单价.Value & "," _
        & "销售日期='" & 销售日期.Value & "'," _
        & "备注='" & 备注.Value & "' " _
        & "where 销售编码='" & 销售编码.Value & "'"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    MsgBox "已经成功将编码为<" & 销售编码.Value & ">的销售信息记录进行修改！", _
        vbInformation, "修改记录"
    '刷新查询和显示
    查询销售信息
    显示销售信息
    myListView
End Sub
```

7.8.8　删除按钮单击事件代码设计

【删除】按钮用于删除当前显示的商品资料记录，程序通过一个删除查询在数据库中完成删除操作。删除操作完成后，程序刷新了窗口显示的记录数据。以下是该过程的详细代码解释：

```
Private Sub 删除记录_Click()
    If MsgBox("本操作将删除编码为<" & 销售编码.Value & ">的销售信息记录！" _
        & vbCrLf & "是否要删除？", _
        vbQuestion + vbYesNo + vbDefaultButton2, "删除记录") = vbNo Then Exit Sub
    '----删除销售信息记录----
    SQL = "delete from 销售信息 where 销售编码='" & 销售编码.Value & "'"
    Set rs = cnn.Execute(SQL)
    MsgBox "已经成功将编码为<" & 销售编码.Value & ">的销售信息记录删除！", _
        vbInformation, "删除记录"
    '刷新查询和显示
    查询销售信息
    显示销售信息
    myListView
End Sub
```

7.8.9　查询按钮单击事件代码设计

在销售信息的查询操作中只能按照销售编码进行查询。单击【查询】按钮后，弹出一个销售编码输入框，在此输入销售编码后，程序从数据库中获取销售编码为用户输入结果的所有销售信息，并将查询到的记录信息显示到窗口中。如图 7-40 所示的是该过程的流程图。

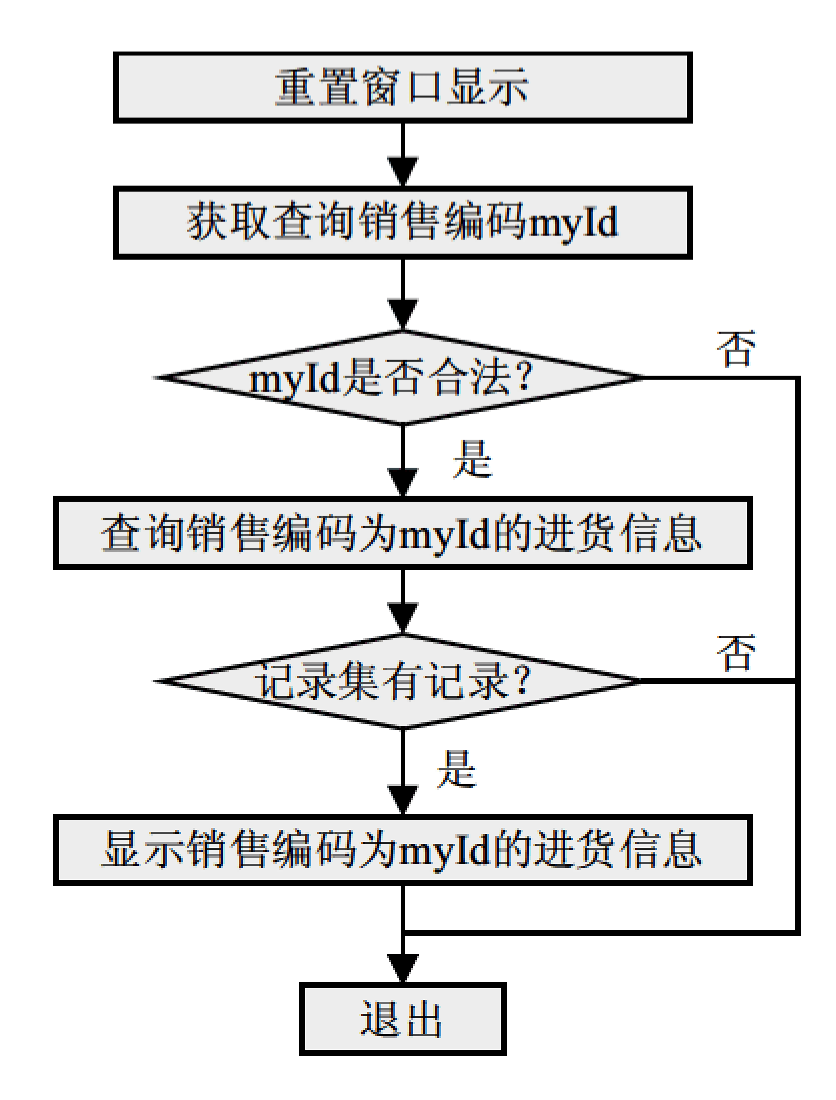


图 7-40　【查询】按钮单击事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 查询记录_Click()
    Dim myId As String
    Dim SQL As String
    Dim i As Integer
    Dim rsSerch As New ADODB.Recordset
    Call 新建记录_Click
    myId = InputBox("请输入销售编码：", "销售查询")
    If Len(Trim(myId)) = 0 Then
        MsgBox "没有输入销售编码！", vbCritical, "警告"
        Exit Sub
    End If
    SQL = "select * from 销售信息 where 销售编码='" & myId & "'"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If rs.BOF And rs.EOF Then
        MsgBox "没有编码为<" & myId & ">的销售信息!", vbCritical, "查询结果"
    Else
        Call 显示销售信息
        销售编码.Enabled = False
    End If
End Sub
```

7.8.10 ListView 控件项目单击事件代码设计

用户在 ListView 控件中单击某个项目后，程序要将该项目的所有数据显示到窗口上部对应的文本框中。程序首先将各个文本框的值清空，然后通过 For 循环，使用 ListView 控件的 ListItems 属性定位项目里各个子项目，并把这些子项目的数据显示到窗口上面对应的文本框中。以下是该过程的详细代码解释：

```
Private Sub ListView1_ItemClick(ByVal Item As MSComctlLib.ListItem)
    On Error Resume Next
    Dim i As Integer
    Call 新建记录_Click
    销售编码.Enabled = False
    Me.Controls(myArray(0)).Value = ListView1.ListItems(Item.Index)
    For i = 1 To UBound(myArray)
        Me.Controls(myArray(i)).Value = ListView1.ListItems(Item.Index).SubItems(i)
    Next
End Sub
```

7.8.11 查询与显示销售信息过程代码设计

查询与显示销售信息两个过程共同完成查询记录集并将记录集数据显示到窗口的工作。查询销售信息首先生成一个查询销售信息字符串，然后按照该查询字符串打开记录集，从而获取对应查询信息。显示进货信息过程使用在查询进货信息过程中获取的供货商信息记录集，

将该记录集首条记录数据写入到窗口的各个对应文本框中。以下是这两个过程的详细代码解释：

```
Public Sub 查询销售信息()
    SQL = "select * from 销售信息 order by 销售编码"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
End Sub

Public Sub 显示销售信息()
    On Error Resume Next
    Dim i As Integer
    Call 新建记录_Click
    '显示第一个销售信息
    rs.MoveFirst
    For i = 0 To UBound(myArray)
        If IsNull(rs.Fields(i)) Then
            Me.Controls(myArray(i)).Value = ""
        Else
            Me.Controls(myArray(i)).Value = rs.Fields(i)
        End If
    Next i
End Sub
```

7.8.12　myListView 过程代码设计

myListView 过程完成 ListView 控件的显示设置以及标题显示任务，然后程序将记录集 rs 的所有记录数据显示到控件上。如图 7-41 所示的是该过程的流程图。

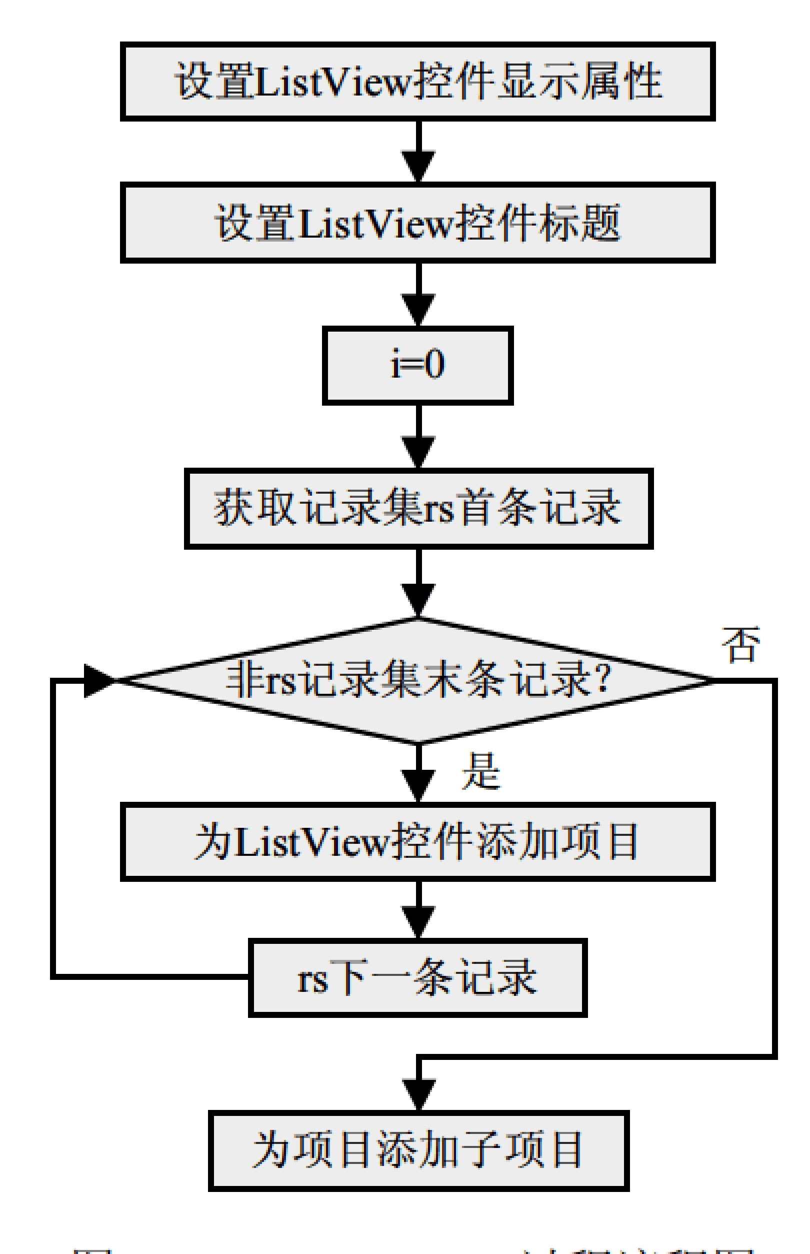


图 7-41　myListView 过程流程图

以下是该过程的详细代码解释：

```
Private Sub myListView()
    On Error Resume Next
    Dim i As Integer
    '设置 ListView 的标题
    With ListView1
        .ColumnHeaders.Clear
        .ListItems.Clear
        .View = lvwReport
        .FullRowSelect = True
        .Gridlines = True
        .Sorted = True
        .ColumnHeaders.Add , , myArray(0)
        For i = 1 To UBound(myArray)
            .ColumnHeaders.Add , , myArray(i)
        Next i
        '设置 ListView 的各行数据
        i = 0
        rs.MoveFirst
        Do While Not rs.EOF
            .ListItems.Add , , rs.Fields(0).Value
            For j = 1 To rs.Fields.Count -1
                .ListItems(i + 1).SubItems(j) = rs.Fields(j).Value
            Next j
            rs.MoveNext
            i = i + 1
        Loop
    End With
    rs.MoveFirst
    销售记录.Caption = "目前数据库中共有 " & i & " 条销售记录"
End Sub
```

7.9　销售统计分析窗体设计

销售统计分析主要对商品的销售情况进行汇总，将汇总数据输出到一个新工作簿中，并根据数据产生销售数据分析图。销售统计分析窗体中涉及到的项目包括统计起止时间、商品名称和商品规格。该窗体的界面如图 7-42 所示。



图 7-42　销售统计分析界面

7.9.1　窗口初始化与关闭事件代码设计

当窗口首次打开时，窗口初始化过程用于对窗口中的控件进行各项初始化设置。在窗口

中的各个复合框都需要在窗口初始化时为控件添加项目。窗口中一共包含了 4 个复合框，其中，两个日期复合框被初始化为系统的开启日期。商品名称和商品规格从销售数据表中获取非一致的记录。如果销售表没有任何记录，这两个复合框将不会被初始化任何值。该过程的流程图如图 7-43 所示。

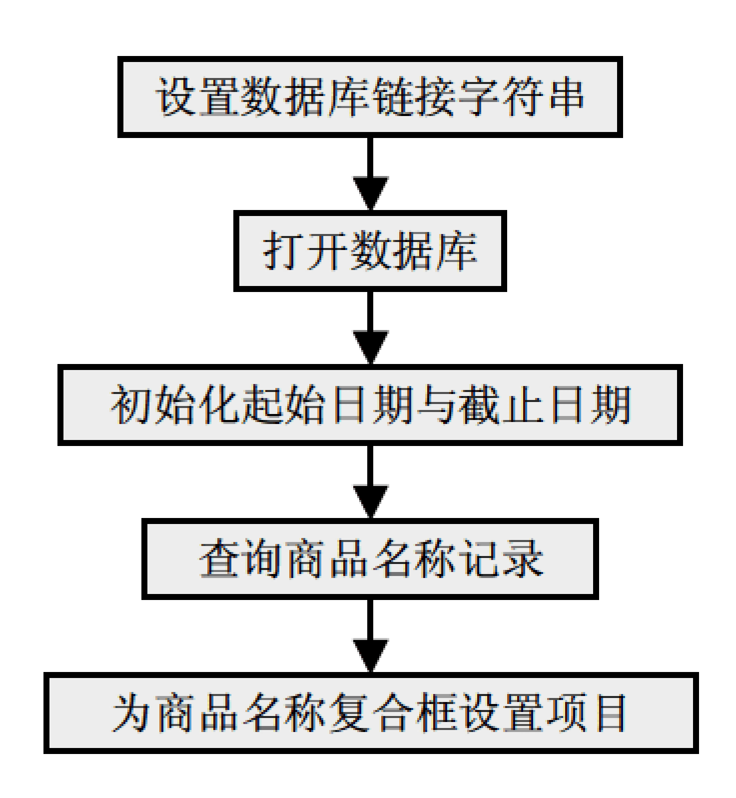


图 7-43　销售统计分析窗口初始化流程图

以下是该过程以及其调用的两个过程的详细代码解释：

```
Dim cnn As New ADODB.Connection
Dim rs As ADODB.Recordset

Private Sub UserForm_Initialize()
    Dim i As Integer
    Dim SQL As String
    '建立与数据库的连接
    With cnn
        .ConnectionString = "Provider=microsoft.jet.oledb.4.0;" _
        & "Data Source=" & ThisWorkbook.Path & "\进销存数据库.mdb;" _
        & "Jet Oledb:database password=123456;"        '设置数据库链接字符串
        .Open                                          '打开数据库
    End With
    '设置时间默认值
    起始日期.Value = Date
    截止日期.Value = Date
    '查询销售信息数据表中不重复的商品名称
    查询商品名称
    '为商品名称复合框设置项目
    商品名称复合框
End Sub

Private Sub 关闭退出_Click()
    rs.Close
    cnn.Close
    Set rs = Nothing
    Set cnn = Nothing
```

```
    Unload 销售统计分析
End Sub
```

7.9.2 查询商品名称过程代码设计

查询商品名称过程从数据库的销售信息表中获取商品名称记录集，该记录集在窗口的其他一些过程中会被调用。以下是该过程的详细代码解释：

```
Public Sub 查询商品名称()
    Dim SQL As String
    Set rs = New ADODB.Recordset
    SQL = "select distinct 商品名称 from 销售信息  where 销售日期 between '" _
        & 起始日期.Value & "' and '" & 截止日期.Value & "'"           '设置查询字符串
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic                  '获取记录集
End Sub
```

7.9.3 商品名称复合框过程代码设计

商品名称复合框过程用于重置窗口的商品名称复合框的项目。复合框中所有的项目都源于查询商品名称过程中获取的商品名称字段记录。该过程通过一个 Do 循环将记录集中所有商品名称信息添加到复合框的项目中。以下是该过程的详细代码解释：

```
Public Sub 商品名称复合框()
    On Error Resume Next
    With 商品名称
        .Clear                                      '清除商品名称复合框项目
        Do While Not rs.EOF                         '循环记录集所有记录
            .AddItem rs.Fields("商品名称")          '为商品名称复合框添加项目
            rs.MoveNext                             '移动到记录集下一条记录
        Loop
    End With
    商品名称.ListIndex = 0                          '设置商品名称复合框默认项目
End Sub
```

7.9.4 复合框事件代码设计

在窗口中有 3 个复合框都具有改变事件，分别是起始日期复合框、截止日期复合框和商品名称复合框。两个日期复合框内容发生变化时，需要检查起始日期和截止日期间的相互关系，并且当用户设置完日期后需要重新刷新选定日期范围内的商品名称项目。商品名称复合框内容发生变更时，需要根据所选商品名称更新商品规格复合框的项目。

两个日期复合框的改变事件相差不大，且这两个事件过程的流程十分简单，以下只给出商品名称复合框改变事件过程的流程图，如图 7-44 所示。

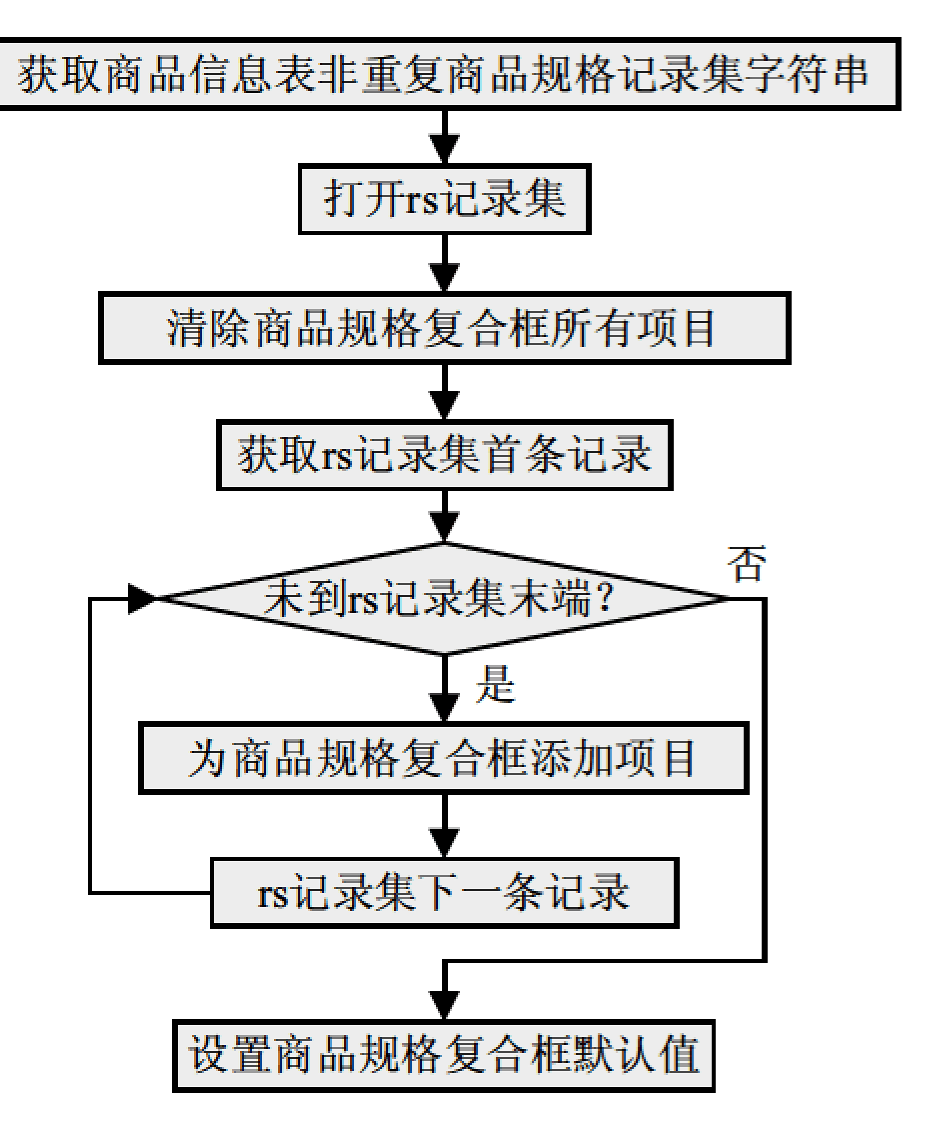


图 7-44　商品名称复合框改变事件流程图

以下是这几个事件过程的详细代码解释：

```
Private Sub 起始日期_Change()
    If 起始日期.Value > 截止日期.Value Then
        MsgBox "起始日期不能大于截止日期！", vbCritical, "警告"
        Exit Sub
    End If
    '查询销售信息数据表中不重复的商品名称
    查询商品名称
    '为商品名称复合框设置项目
    商品名称复合框
End Sub

Private Sub 截止日期_Change()
    If 截止日期.Value < 起始日期.Value Then
        MsgBox "截止日期不能小于起始日期！", vbCritical, "警告"
        Exit Sub
    End If
    '查询销售信息数据表中不重复的商品名称
    查询商品名称
    '为商品名称复合框设置项目
    商品名称复合框
End Sub

Private Sub 商品名称_Change()
    On Error Resume Next
    Dim SQL As String
    '查询指定商品名称下的不重复规格名称
    SQL = "select distinct 商品规格 from 商品信息 " _
        & "where 商品名称='" & 商品名称.Value & "'"
    Set rs = New ADODB.Recordset
```

```
        rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
        '为商品规格复合框设置项目
        With 商品规格
            .Clear
            Do While Not rs.EOF
                .AddItem rs.Fields("商品规格")
                rs.MoveNext
            Loop
        End With
        商品规格.ListIndex = 0
End Sub
```

7.9.5 按钮单击事件代码设计

在该窗口中一共包含了两个按钮，其中统计分析按钮的单击事件代码比较复杂，以下将重点介绍。重置条件按钮用于重置窗口中几个复合框的项目。统计分析过程按照用户设置的查询条件查询销售结果，然后将该结果做统计分析工作并将结果保存到新工作簿中。

统计分析过程首先创建新的工作簿，然后在工作簿的工作表中设置标题以及统计表的表头。随后程序从数据库获取数据填充到工作表的对应各列中，最后程序根据刚获取的数据生成图表。该过程的流程图如图 7-45 所示。

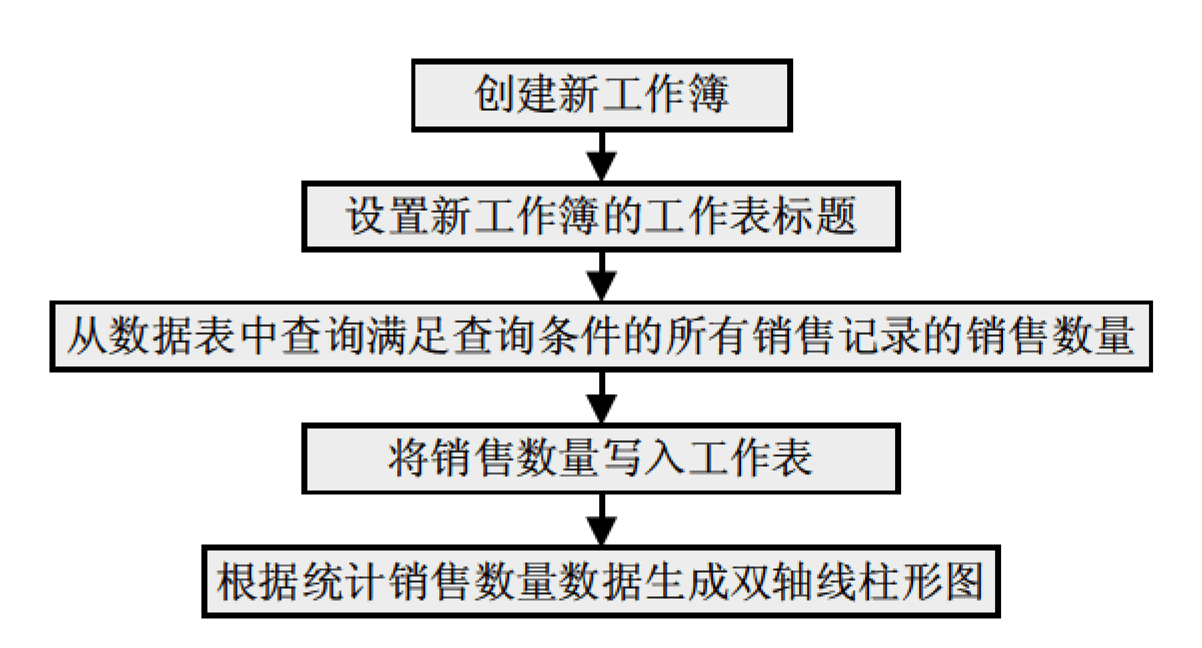


图 7-45　统计分析按钮单击事件过程流程图

以下是两个按钮单击事件过程的详细代码解释：

```
Private Sub 重置条件_Click()
        On Error Resume Next
        起始日期.Value = Date
        截止日期.Value = Date
        商品名称.ListIndex = 0
        商品规格.ListIndex = 0
End Sub

Private Sub 统计分析_Click()
        Dim i As Integer
        Dim myDate As Date
        Dim SQL As String
        Dim rsx As ADODB.Recordset
        Dim wb As Workbook
        Dim ws As Worksheet
        '创建新工作簿
        Set wb = Workbooks.Add
        Set ws = wb.ActiveSheet
```

```
'设置工作表标题
ws.Range("A1") = 商品名称.Value & " 的销售统计"
With ws.Range("A1:E1")
    .Merge
    .Font.Size = 15
    .RowHeight = 20
    .Font.Bold = True
    .HorizontalAlignment = xlCenter
    .Borders(xlEdgeBottom).Weight = xlMedium
End With
ws.Range("A2:E2") = Array("日期", "商品名称", "商品规格", "销售量", "销售额")
'从数据表中查询数据
i = 3
For myDate = 起始日期.Value To 截止日期.Value
    ws.Range("A" & i) = Format(myDate, "yyyy-mm-dd")
    ws.Range("B" & i) = 商品名称.Value
    ws.Range("C" & i) = 商品规格.Value
    '计算某日指定规格之商品的销售总数和销售总额
    If 商品规格.Value = "全部规格" Then
        SQL = "select sum(销售数量) as aa," _
            & "sum(销售数量*销售单价) as bb from 销售信息 " _
            & "where 销售日期='" & myDate & "'" _
            & " and 商品名称='" & 商品名称.Value & "'"
    Else
        SQL = "select sum(销售数量) as aa," _
            & "sum(销售数量*销售单价) as bb from 销售信息 " _
            & "where 销售日期='" & myDate & "'" _
            & " and 商品名称='" & 商品名称.Value & "'" _
            & " and 商品规格='" & 商品规格.Value & "'"
    End If
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenStatic, adLockOptimistic
    If IsNull(rs!aa) Then
        ws.Range("D" & i) = 0
    Else
        ws.Range("D" & i) = rs!aa
    End If
    If IsNull(rs!bb) Then
        ws.Range("E" & i) = 0
    Else
        ws.Range("E" & i) = rs!bb
    End If
    i = i + 1
Next
ws.Columns.AutoFit
'开始绘制统计分析图表
Range("A2:E6").Select
Charts.Add
ActiveChart.ApplyCustomType ChartType:=xlBuiltIn, TypeName:="两轴线-柱图"
ActiveChart.SetSourceData Source:= _
```

```
        Sheets("Sheet1").Range("A2:A" & i -1 & ",D2:E" & i -1), PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:="Sheet1"
    With ActiveChart
        .HasTitle = True
        .ChartTitle.Characters.Text = 商品名称.Value & " 销售统计图表"
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "日期"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "销售量"
        .Axes(xlCategory, xlSecondary).HasTitle = False
        .Axes(xlValue, xlSecondary).HasTitle = True
        .Axes(xlValue, xlSecondary).AxisTitle.Characters.Text = "销售额(元)"
    End With
    Set ws = Nothing
    Set wb = Nothing
End Sub
```

7.10 库存管理模块设计

库存管理模块通过库存管理窗口实现，通过该窗口用户可以查看各个商品的实际库存量。该窗口只需要实现库存商品信息的显示功能，因此结构并不复杂，包含的过程数量并不多。

7.10.1 库存资料管理窗体设计

库存资料管理窗体包含的控件数量十分少，包含一个框架控件、一个 ListView 控件、一个标签控件和一个按钮控件。框架控件用于包含 ListView 控件，ListView 控件用于显示库存记录信息，标签用于显示提示信息，按钮用于退出窗口。如图 7-46 所示是该窗口的界面。



图 7-46　库存资料管理界面

7.10.2　窗口初始化过程代码设计

在库存窗口被显示时，为确保每次显示的库存数据都是更新后的数据，程序需要重新获取商品的所有销售数据和进货数据，然后根据这些数据确定实际的商品库存量，最后将这些数据显示到窗口中并保存到数据库库存信息表。该过程的流程如图 7-47 所示。

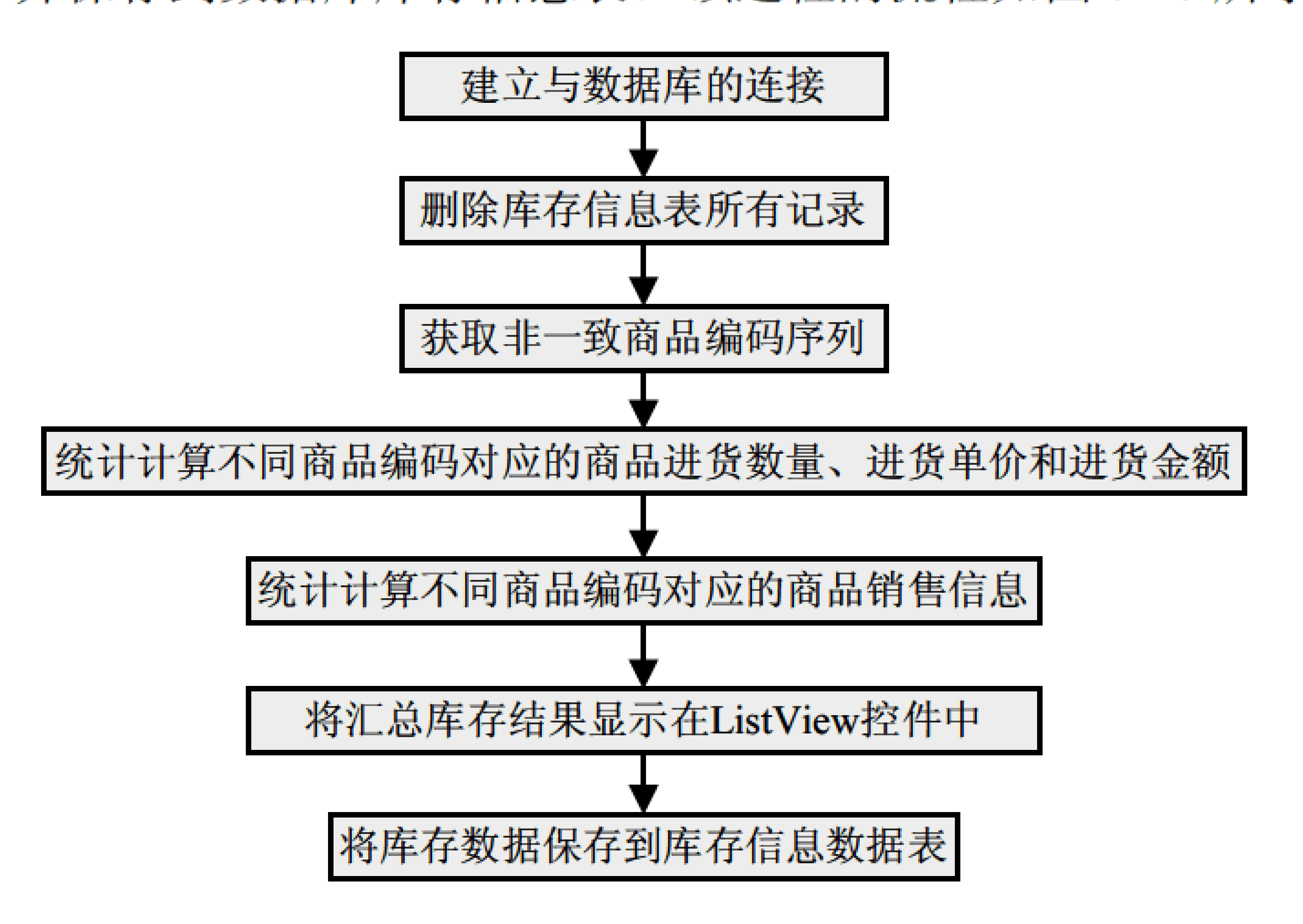


图 7-47　库存窗口初始化过程流程图

以下是该过程的详细代码解释：

```
Dim myArray As Variant
Dim cnn As New ADODB.Connection
Dim rs As ADODB.Recordset

Private Sub UserForm_Initialize()
    Dim i As Integer
    Dim n As Long
    myArray = Array("商品编码", "商品名称", "商品规格", "计量单位", _
        "库存数量", "库存单价", "库存金额")
    '建立与数据库的连接
    With cnn
        .ConnectionString = "Provider=microsoft.jet.oledb.4.0;" _
        & "Data Source=" & ThisWorkbook.Path & "\进销存数据库.mdb;" _
        & "Jet Oledb:database password=123456;"
        .Open
    End With
    '从进货信息数据表和销售信息数据表中查询记录，统计商品库存信息
    '先删除库存信息数据表中的所有记录
    SQL = "delete from  库存信息"
    Set rs = cnn.Execute(SQL)
    '查询进货信息数据表中不重复的商品编码
    SQL = "select distinct  商品编码  from  进货信息"
```

```
Set rs = New ADODB.Recordset
rs.Open SQL, cnn, adOpenStatic, , dCmdText
n = rs.RecordCount
If n <= 0 Then Exit Sub
ReDim myCode(1 To n) As String
ReDim myName(1 To n) As String
ReDim mySpec(1 To n) As String
ReDim myUnit(1 To n) As String
ReDim myprice(1 To n) As Single
ReDim myNum(1 To n, 1 To 2) As Integer
ReDim myMoney(1 To n, 1 To 2) As Single
For i = 1 To n
    myCode(i) = rs.Fields("商品编码")
    rs.MoveNext
Next
'统计计算不同商品编码对应的商品进货数量、进货单价和进货金额
For i = 1 To n
    SQL = "select 商品名称,商品规格,计量单位,进货单价 from 进货信息" _
        & " where 商品编码='" & myCode(i) & "'"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    myName(i) = rs!商品名称
    mySpec(i) = rs!商品规格
    myUnit(i) = rs!计量单位
    myprice(i) = rs!进货单价
    SQL = "select sum(进货数量) as 数量," _
        & " sum(进货数量*进货单价) as 金额 from 进货信息" _
        & " where 商品编码='" & myCode(i) & "'"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    myNum(i, 1) = rs!数量
    myMoney(i, 1) = rs!金额
Next
'统计计算不同商品编码对应的商品销售信息
For i = 1 To n
    SQL = "select sum(销售数量) as 数量," _
        & " sum(销售数量*销售单价) as 金额 from 销售信息" _
        & " where 商品编码='" & myCode(i) & "'"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If IsNull(rs!数量) Then
        myNum(i, 2) = 0
    Else
        myNum(i, 2) = rs!数量
    End If
    If IsNull(rs!金额) Then
        myMoney(i, 2) = 0
```

```
        Else
            myMoney(i, 2) = rs!金额
        End If
    Next
    '将汇总计算结果显示在 ListView 控件中
    With ListView1
        '设置 ListView 的标题
        .ColumnHeaders.Clear
        .ListItems.Clear
        .View = lvwReport
        .FullRowSelect = True
        .Gridlines = True
        .Sorted = True
        .ColumnHeaders.Add , , myArray(0)
        For i = 1 To UBound(myArray)
            .ColumnHeaders.Add , , myArray(i)
        Next i
        '设置 ListView 的各行数据
        For i = 1 To n
            .ListItems.Add , , myCode(i)
            .ListItems(i).SubItems(1) = myName(i)
            .ListItems(i).SubItems(2) = mySpec(i)
            .ListItems(i).SubItems(3) = myUnit(i)
            .ListItems(i).SubItems(4) = myNum(i, 1) -myNum(i, 2)
            If myNum(i, 1) -myNum(i, 2) = 0 Then
                .ListItems(i).SubItems(5) = 0
                .ListItems(i).SubItems(6) = 0
            Else
                .ListItems(i).SubItems(5) = myprice(i)
                .ListItems(i).SubItems(6) = myMoney(i, 1) -myMoney(i, 2)
            End If
        Next i
    End With
    库存记录.Caption = "目前数据库中共有 " & n & " 条库存记录"
    '下面将库存数据保存到库存信息数据表
    '首先删除库存信息数据表的所有记录
    SQL = "delete from 库存信息"
    Set rs = cnn.Execute(SQL)
    '将新的库存信息保存到库存信息数据表
    Set rs = New ADODB.Recordset
    rs.Open "库存信息", cnn, adOpenKeyset, adLockOptimistic
    For i = 1 To n
        rs.AddNew
        rs.Fields("商品编码") = myCode(i)
        rs.Fields("商品名称") = myName(i)
        rs.Fields("商品规格") = mySpec(i)
        rs.Fields("计量单位") = myUnit(i)
```

```
            rs.Fields("库存数量") = myNum(i, 1) -myNum(i, 2)
            If myNum(i, 1) -myNum(i, 2) = 0 Then
                rs.Fields("库存单价") = 0
                rs.Fields("库存金额") = 0
            Else
                rs.Fields("库存单价") = myprice(i)
                rs.Fields("库存金额") = myMoney(i, 1) -myMoney(i, 2)
            End If
            rs.Update
        Next i
End Sub
```

7.10.3 关闭退出按钮代码设计

关闭退出按钮用于退出窗口，在退出前该按钮单击事件还需要清除一些临时变量。以下是该过程的代码：

```
Private Sub 关闭退出_Click()
    rs.Close
    cnn.Close
    Set cnn = Nothing
    Set rs = Nothing
    Unload 库存资料管理
End Sub
```

7.11 资料查询与导出

资料查询与导出窗口将系统中有关数据查询与导出的工作集中完成。在前面所介绍的进销存 3 个模块中分别都包含了部分的查询与导出功能，但是这些功能都只涉及到对应模块下的查询与导出功能。在本模块中，用户可以完成所有功能模块的查询与导出工作。

7.11.1 资料查询与导出窗体设计

在本查询与导出窗口中，用户一共可以设置 3 个查询设置条件。当用户在设置运算符选择了 between 时，用户还可以设置两个条件值。而选择其他的运算符时，只需要设置一个查询值就可以了。在窗口的右侧包含了 4 个功能按钮，分别是重设条件、开始查询、数据导出和关闭窗口。

重设条件按钮用于重置窗口中所有复合框数据，以便于用户重新输入查询数据。开始查询按钮将使用用户当前设置的查询条件获取查询结果，并将查询结果显示在窗口底部的 ListView 控件中。数据导出按钮用于将查询所得数据导出到一个新的工作表中。该窗口的界面如图 7-48 所示。

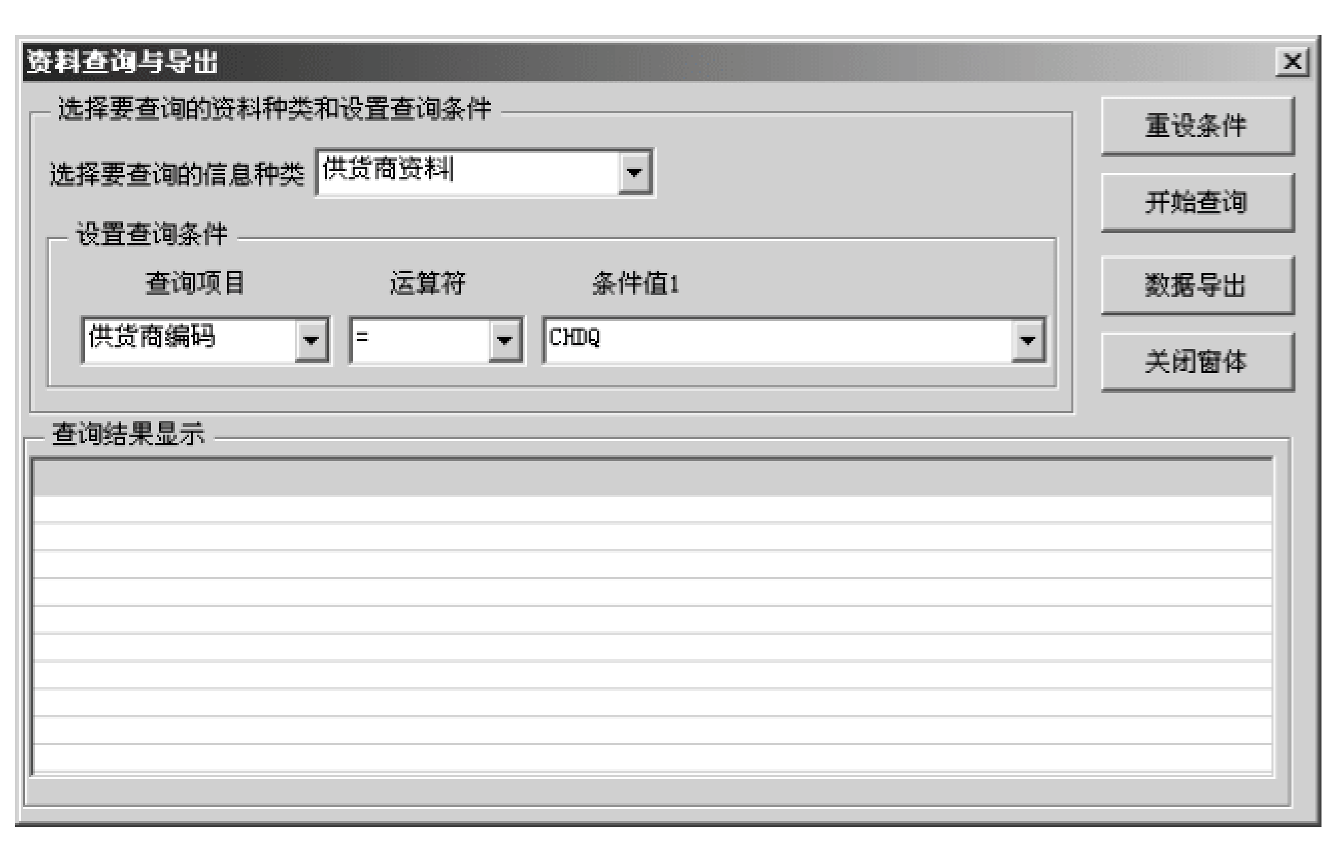


图 7-48　资料查询与导出界面

7.11.2　窗口初始化与关闭过程代码设计

窗口初始化时，需要完成的设置工作比较多，程序需要建立到数据库的链接、为窗口中各个复合框设置项目以及重置查询结果显示。如图 7-49 所示是该过程的流程图。

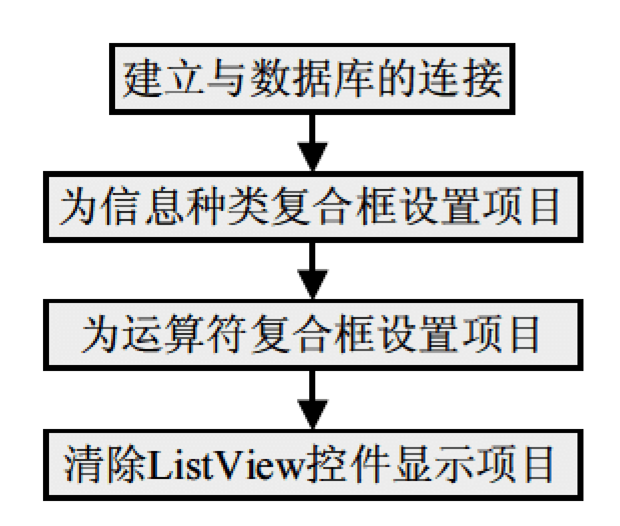


图 7-49　资料查询与导出窗口初始化流程图

以下是窗口初始化与关闭事件的详细代码解释：

```
Dim myTable As String
Dim cnn As New ADODB.Connection
Dim rs As ADODB.Recordset
Private Sub UserForm_Initialize()
    '建立与数据库的连接
    With cnn
        .ConnectionString = "Provider=microsoft.jet.oledb.4.0;" _
        & "Data Source=" & ThisWorkbook.Path & "\进销存数据库.mdb;" _
        & "Jet Oledb:database password=123456;"
        .Open
    End With
    '为信息种类复合框设置项目
    With 信息种类
        .AddItem "商品资料"
        .AddItem "商品资料"
        .AddItem "进货资料"
```

```
            .AddItem "销售资料"
            .AddItem "库存资料"
    End With
    信息种类.ListIndex = 0
    Select Case 信息种类.Value
        Case "商品资料"
            myTable = "商品信息"
        Case "商品资料"
            myTable = "商品信息"
        Case "进货资料"
            myTable = "进货信息"
        Case "销售资料"
            myTable = "销售信息"
        Case "库存资料"
            myTable = "库存信息"
    End Select
    '为运算符复合框设置项目
    With 运算符
        .AddItem "="
        .AddItem ">"
        .AddItem "<"
        .AddItem ">="
        .AddItem "<="
        .AddItem "<>"
        .AddItem "like"
        .AddItem "between"
    End With
    运算符.ListIndex = 0
    清除显示信息
End Sub

Private Sub 关闭窗体_Click()
    cnn.Close
    Set rs = Nothing
    Set cnn = Nothing
    Unload 资料查询与导出
End Sub
```

7.11.3 查询项目复合框代码设计

用户在查询项目复合框中选择了某项目后，对应的条件值会随该值发生改变。程序在查询项目复合框的改变事件中实现该操作：首先从数据库中获取非一致查询项目记录集，然后通过两个 Do 循环将这些记录数据添加到条件值 1 和条件值 2 两个复合框中。如图 7-50 所示是该过程的流程图。

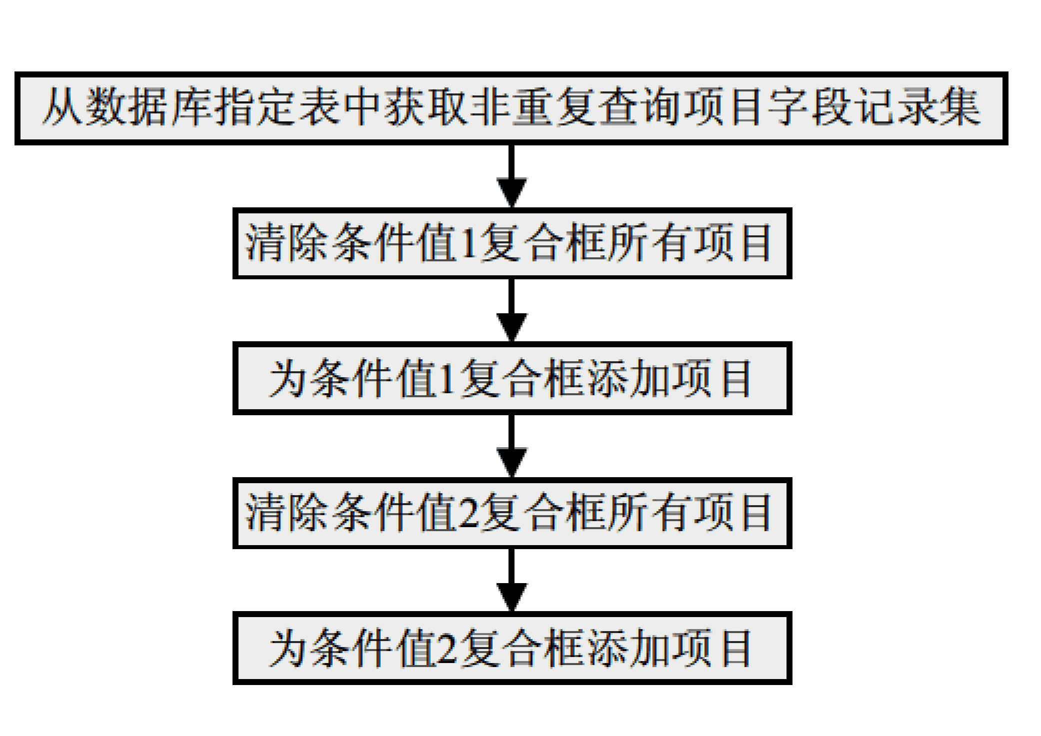


图 7-50　查询项目复合框改变事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 查询项目_Change()
    On Error GoTo hhh
    Dim i As Integer
    Set rs = New ADODB.Recordset
    SQL = "select distinct " & 查询项目.Value & " from " & myTable
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    With 条件值 1
        .Clear
        i = 1
        Do While Not rs.EOF
            .AddItem rs.Fields(查询项目.Value)
            rs.MoveNext
            i = i + 1
        Loop
    End With
    条件值 1.ListIndex = 0
    rs.MoveFirst
    With 条件值 2
        .Clear
        i = 1
        Do While Not rs.EOF
            .AddItem rs.Fields(查询项目.Value)
            rs.MoveNext
            i = i + 1
        Loop
    End With
    条件值 2.ListIndex = 0
hhh:
    清除显示信息
End Sub
```

7.11.4　开始查询按钮代码设计

【开始查询】按钮使用用户当前进行的设置查询数据库，将获取的记录集显示到窗口的

查询结果显示控件中。程序通过一个 Select 查询语句获取查询记录，该 Select 查询语句查询的工作表 myTable 由窗口初始化过程确定。查询条件设置过程较为复杂。按照运算方式的不同，查询条件的格式分为 3 种：为 between 时，需要设置两个查询条件值；为 Like 时，需要设置模糊查询格式；其他的查询条件下直接使用运算符即可。另外，查询项目不一样时，需要比较的值也会不一样。如图 7-51 所示的是该过程执行流程图。

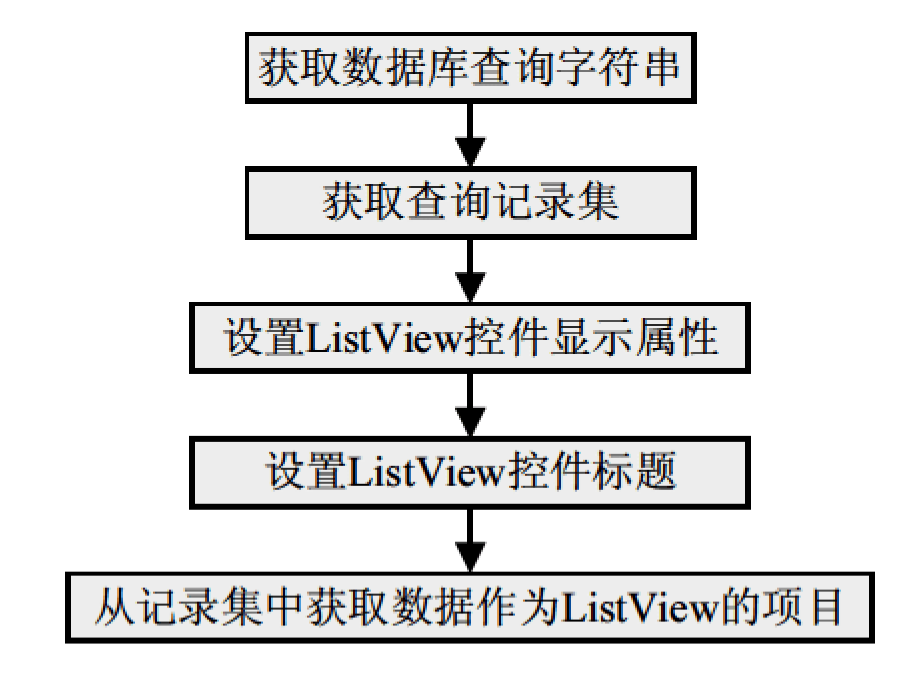


图 7-51 【开始查询】按钮单击事件流程图

以下是该过程的详细代码解释：

```
Private Sub 开始查询_Click()
    Dim SQL As String
    Dim Condition As String, Con0 As String, Con1 As String, Con2 As String
    '设置查询条件
    Con0 = " where "
    If 查询项目.Value = "进货日期" Or 查询项目.Value = "销售日期" Then
        Con1 = "'" & Format(条件值 1.Value, "yyyy-mm-dd") & "'"
        Con2 = "'" & Format(条件值 2.Value, "yyyy-mm-dd") & "'"
    ElseIf 查询项目.Value = "最高库存" Or 查询项目.Value = "最低库存" _
        Or 查询项目.Value = "进货数量" Or 查询项目.Value = "进货单价" _
        Or 查询项目.Value = "销售数量" Or 查询项目.Value = "销售单价" _
        Or 查询项目.Value = "库存数量" Or 查询项目.Value = "库存单价" _
        Or 查询项目.Value = "库存金额" Then
        Con1 = Val(条件值 1.Value)
        Con2 = Val(条件值 2.Value)
    Else
        Con1 = "'" & 条件值 1.Value & "'"
        Con2 = "'" & 条件值 2.Value & "'"
    End If
    Condition = " where " & 查询项目.Value
    If 运算符.Value = "between" Then
        Condition = Condition & " between " & Con1 & " and " & Con2
    ElseIf 运算符.Value = "like" Then
        Condition = Condition & " like '%" & 条件值 1.Value & "%'"
    Else
        Condition = Condition & 运算符.Value & Con1
    End If
    '设置 SQL 语句
```

```
    SQL = "select * from " & myTable & Condition
    '开始查询
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If rs.BOF And rs.EOF Then
        MsgBox "没有查询到结果！ ", vbCritical, "查询结果"
        清除显示信息
        Exit Sub
    End If
    '将查询结果显示在 ListView 控件中
    With ListView1
        '设置 ListView1 的标题、显示类型、整行选择和网格线属性
        .ColumnHeaders.Clear
        .ListItems.Clear
        .View = lvwReport
        .FullRowSelect = True
        .Gridlines = True
        '为 ListView1 设置标题
        For i = 0 To rs.Fields.Count -1
            .ColumnHeaders.Add , , rs.Fields(i).Name
        Next
        '为 ListView1 设置各行数据
        i = 1
        .ListItems.Clear
        Do While Not rs.EOF
            .ListItems.Add , , rs.Fields(0).Value
            For j = 1 To rs.Fields.Count -1
                .ListItems(i).SubItems(j) = rs.Fields(j).Value
            Next j
            rs.MoveNext
            i = i + 1
        Loop
        rs.MoveFirst
    End With
End Sub
```

7.11.5 数据导出按钮代码设计

【数据导出】按钮将用户查询获取结果导出到工作表中。程序首先从数据表中获取字段名称作为工作表的列标题，然后调整标题的格式，最后从数据记录集中获取数据填充到工作表中完成导出工作。该过程流程图如图 7-52 所示。

以下是该过程的详细代码解释：

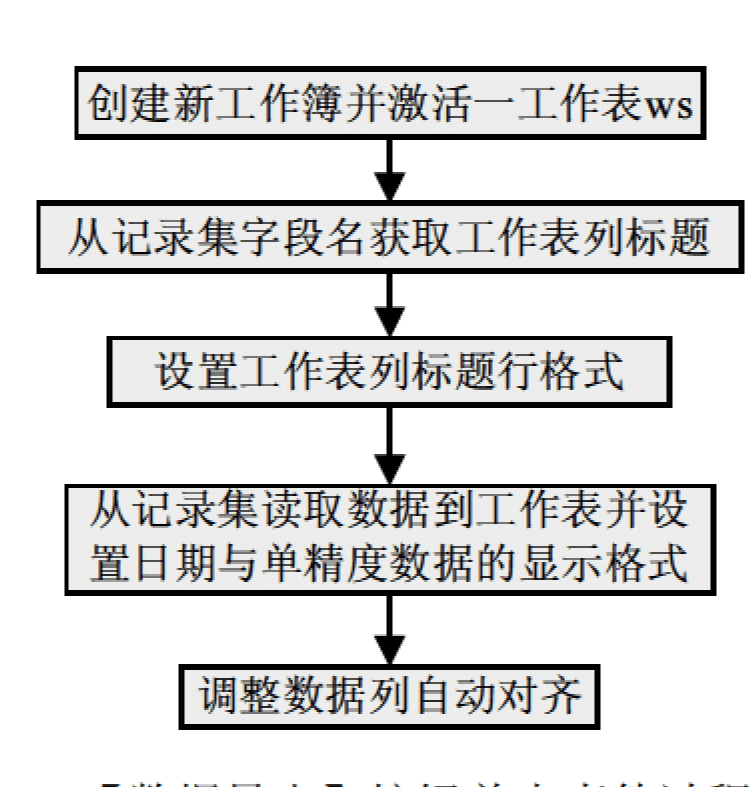


图 7-52 【数据导出】按钮单击事件过程流程图

```
Private Sub 数据导出_Click()
    Dim wb As Workbook
    Dim ws As Worksheet
    Dim i As Integer, j As Integer
    Set wb = Workbooks.Add
    Set ws = wb.ActiveSheet
    With ws
        For i = 0 To rs.Fields.Count -1
            .Cells(1, i + 1) = rs.Fields(i).Name
        Next
        With .Range(Cells(1, 1), Cells(1, rs.Fields.Count))
            .Font.Bold = True
            .HorizontalAlignment = xlCenter
        End With
        i = 1
        Do While Not rs.EOF
            For j = 0 To rs.Fields.Count -1
                .Cells(i + 1, j + 1) = rs.Fields(j)
                If rs.Fields(j).Type = 135 Then
                    .Cells(i + 1, j + 1).NumberFormat = "yyyy-mm-dd"
                End If
                If rs.Fields(j).Type = 4 Then
                    .Cells(i + 1, j + 1).NumberFormat = "0.00"
                End If
            Next
            rs.MoveNext
            i = i + 1
        Loop
        .Columns.AutoFit
    End With
    Set ws = Nothing
    Set wb = Nothing
End Sub
```

7.11.6 信息种类复合框代码设计

信息种类复合框中的选项对查询的信息进行了第一级分类。当用户改变该项选择时，查询项目复合框以及 ListView 控件的显示都需要随之刷新。首先，程序通过一个 Select Case 语句确定用户选择的信息种类，该信息种类在数据库中有对应的数据表存在。该数据表中的字段名称将作为查询项目的新项目。其次，程序通过清除显示信息过程清除 ListView 控件中所有的项目。如图 7-53 所示是该过程的流程图。

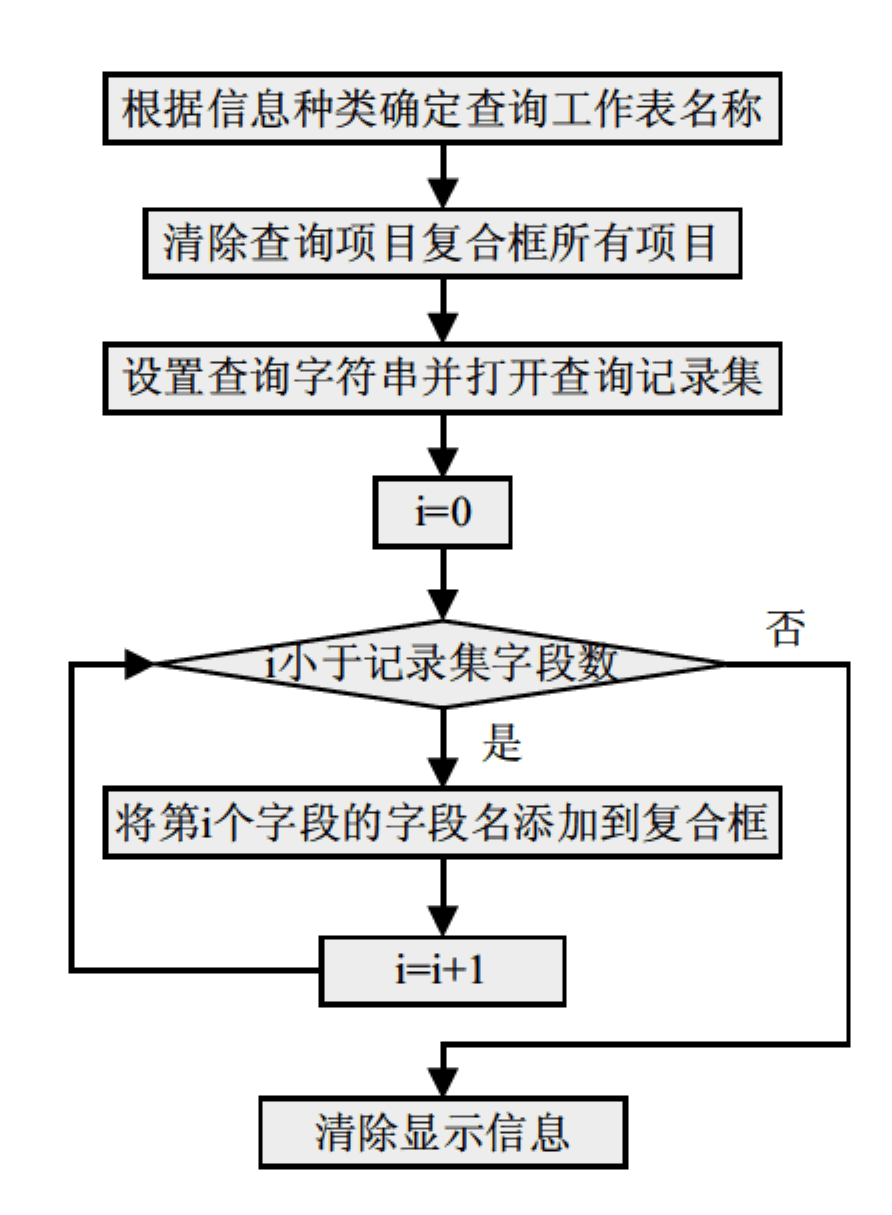


图 7-53　信息种类复合框改变事件过程流程图

以下是该过程的详细代码解释：

```
Private Sub 信息种类_Change()
    On Error Resume Next
    Select Case 信息种类.Value
        Case "商品资料"
            myTable = "商品信息"
        Case "商品资料"
            myTable = "商品信息"
        Case "进货资料"
            myTable = "进货信息"
        Case "销售资料"
            myTable = "销售信息"
        Case "库存资料"
            myTable = "库存信息"
    End Select
    With 查询项目
        .Clear
        Set rs = New ADODB.Recordset
        SQL = "select * from " & myTable
        rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
        For i = 0 To rs.Fields.Count -1
            .AddItem rs.Fields(i).Name
        Next
    End With
    查询项目.ListIndex = 0
    清除显示信息
End Sub
```

7.11.7 运算符复合框事件代码设计

用户在运算符复合框中选择了 between 时，条件值 2 复合框会被设置为可见。此时还要调整条件值 1 和条件值 2 两个复合框的宽度，以使这两个复合框都能正确显示。当再选中其他的运算符时，程序又需要重新设置两个复合框的状态。以下是该功能实现的过程详细代码解释：

```
Private Sub 运算符_Change()
    If 运算符.Value <> "between" Then
        Label_and.Visible = False
        Label_Value2.Visible = False
        条件值 2.Visible = False
        条件值 1.Width = 179
    Else
        Label_and.Visible = True
        Label_Value2.Visible = True
        条件值 2.Visible = True
        条件值 1.Width = 79
    End If
    清除显示信息
End Sub
```

7.11.8 重设条件与清除显示信息代码设计

重设条件按钮被单击时，将重置窗口中所有输入控件以及刷新查询结果显示的数据，其中一共涉及到了窗口中的 5 个复合框控件和 1 个 ListView 控件。刷新 ListView 控件的显示通过调用清除显示信息过程实现。以下是这两个过程的详细代码解释：

```
Private Sub 重设条件_Click()
    信息种类.ListIndex = 0
    查询项目.ListIndex = 0
    运算符.ListIndex = 0
    条件值 1.ListIndex = 0
    条件值 2.ListIndex = 0
    清除显示信息
End Sub

Public Sub 清除显示信息()
    With ListView1
        .ColumnHeaders.Clear
        .ListItems.Clear
        .View = lvwReport
        .FullRowSelect = True
        .Gridlines = True
    End With
End Sub
```

7.12　系统测试

本系统包含 3 个关键部分，分别是商品的进销存操作。在本节系统测试部分不仅包含了这 3 个部分的测试，另外还包含了系统中各项数据的查询测试，以下通过 3 个小节分别展开这些内容。由于库存操作十分简单，而且属于查询功能部分，下面将把这部分内容列入到查询小节中介绍。

7.12.1　进货测试

（1）在加载项菜单中依次选择【系统设置】|【用户登录】命令，在随后弹出的登录窗口中设置【用户名】为【管理员】、【密码】为 1111，如图 7-54 所示，单击【确定】按钮即可。由于系统中的各项功能都有使用权限，为了测试方便，这里使用了管理员来登录，以便直接使用各项功能。



图 7-54　用户登录

（2）在加载项菜单中依次选择【进货管理】|【进货日常管理】命令，在随后弹出的进货资料管理窗口中单击【新建】按钮，然后设置【进货编码】为 JHBM00002，【供货商编码】设为 CHDQ，【商品编码】设为 SPXX00005，设置【进货数量】为 15、【进货单价】为 2000。单击【保存】按钮即可，如图 7-55 所示。

（3）用户成功保存进货资料后，在进货信息清单的列表中随即会刷新该项进货信息。可在该列表中单击该项销售数据查看或编辑该记录，效果如图 7-56 所示。



图 7-55　进货资料管理窗口设置

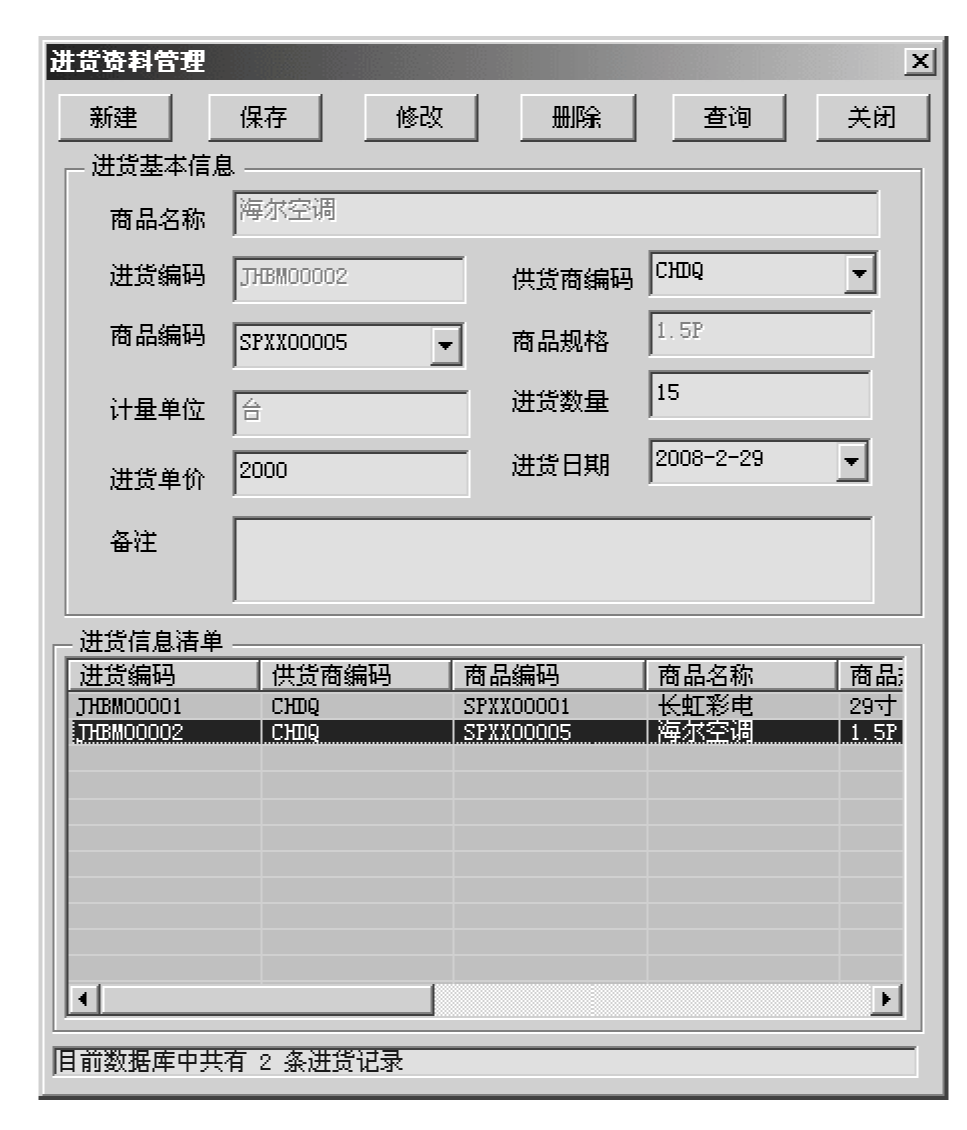


图 7-56　进货资料管理

7.12.2 销售测试

（1）在加载项菜单中依次选择【销售管理】|【销售日常管理】命令，在随后打开的销售资料管理窗口中设置【销售编码】为 XSBM00001、【商品编码】为 SPXX00005，设置【销售数量】为 1、【销售单价】为 3000。单击【保存】按钮即可。设置资料管理界面如图 7-57 所示。

（2）保存完成后，该条销售记录会立即显示在窗口的销售信息清单中。单击该条项目，窗口中的数据会随之刷新显示用户选择销售项目的信息。新建的销售清单信息如图 7-58 所示。

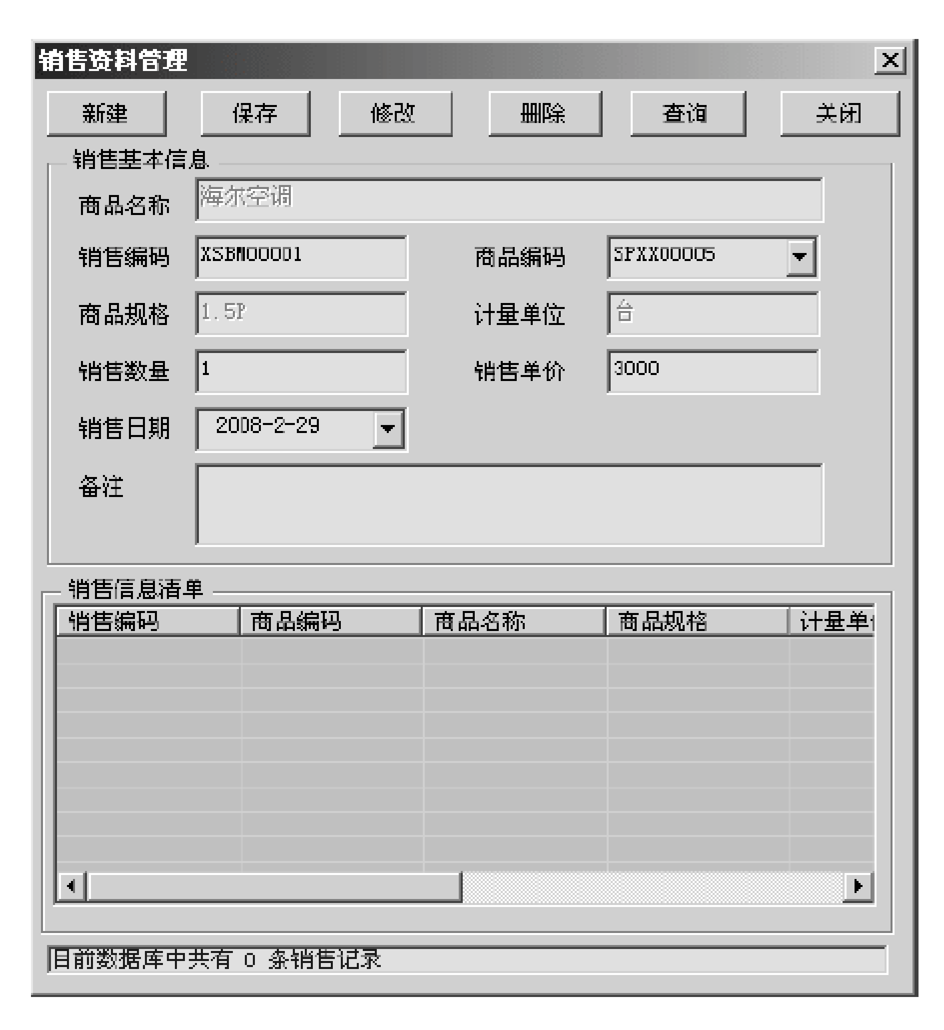


图 7-57 销售资料管理设置

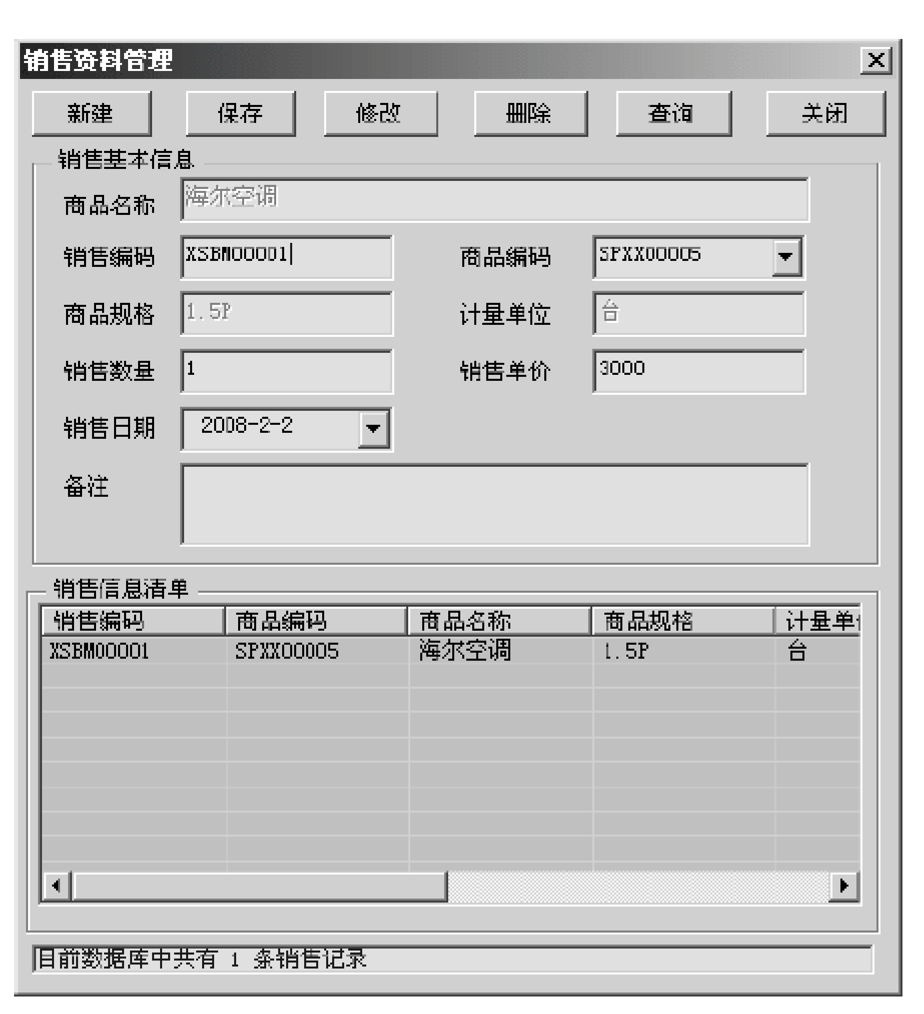


图 7-58 新建的销售清单信息

7.12.3 查询与导出测试

该部分查询以查询库存情况为例。具体操作步骤如下：

（1）在加载项菜单中依次选择【资料查询与导出】|【库存资料查询与导出】命令，在随后打开的查询窗口中设置【查询项目】为“商品编码”、【运算符】为=、【条件值 1】为 SPXX00001，如图 7-59 所示。

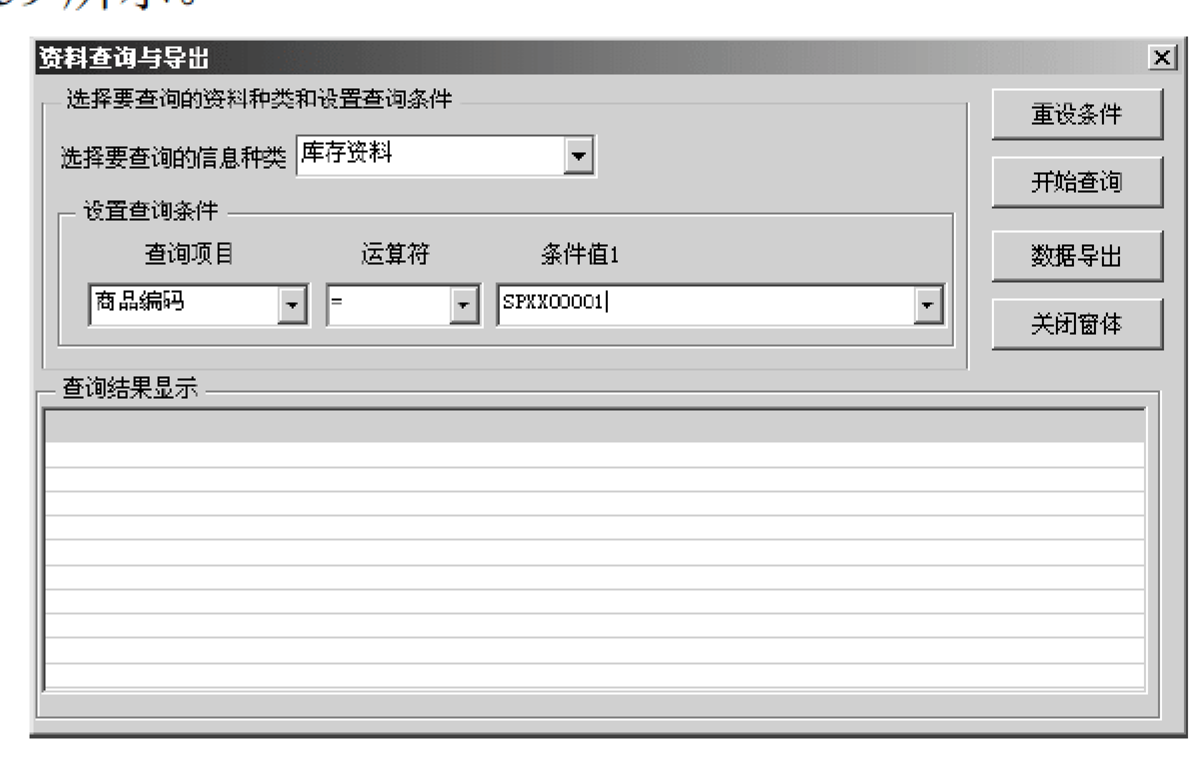
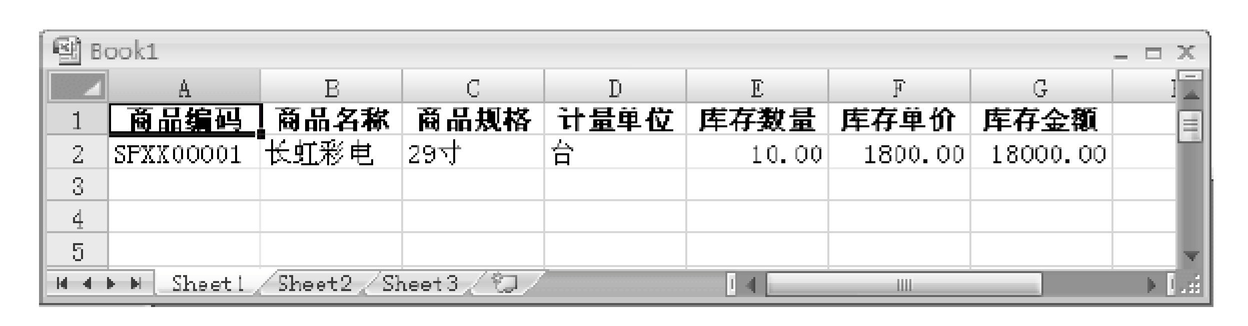


图 7-59 资料查询与导出设置

（2）在窗口中单击【开始查询】按钮，此时窗口的查询结果显示列表中将会显示该查询结果，如图 7-60 所示。单击【数据导出】按钮，查询活动的数据将会被保存到一个新工作表中，如图 7-61 所示。



图 7-60　查询结果显示



	A	B	C	D	E	F	G
1	商品编码	商品名称	商品规格	计量单位	库存数量	库存单价	库存金额
2	SPXX00001	长虹彩电	29寸	台	10.00	1800.00	18000.00

图 7-61　导出查询数据

第 8 章　员工管理系统

员工管理是企业管理的重要内容之一。合理的人事管理可以帮助公司培养一支有纪律、高素质团结合作的队伍，同时也会为公司树立良好形象奠定基础，促进企业发展；而员工管理系统就实现了对员工信息的集中而系统的管理，使得管理者可以更加轻松地完成员工信息的存储与查询操作。

8.1　系 统 概 论

员工管理系统包含员工档案管理与员工考勤管理。员工档案管理主要完成员工资料的建立、编辑和查询操作。员工的考勤管理主要完成签到、请假登记操作，是计算员工工资报酬的重要依据。

8.1.1　设计思路

员工管理系统一共包含了 3 个主要功能，分别对应首页的 3 个按钮，即员工信息资料管理、员工考勤签到和员工请假登记。该系统的架构如图 8-1 所示。

员工管理系统
员工信息资料管理
员工考勤签到
员工请假登记

图 8-1　员工管理系统结构图

员工管理系统包含了主页表、员工档案卡表、请假登记表、考勤表、库表和参数表 6 个表。系统将所有员工的信息存储在一个员工库表中，在其中可以完成员工档案的建立、编辑和查询操作。以下是各个表的详细功能介绍。

- 主页表：该表是系统的首页，完成功能跳转工作。
- 员工档案卡表：用于显示员工档案卡。在该表中可以完成所有与员工档案相关的操作，包括增加、删除、修改、查询、浏览记录等。
- 请假登记表：该表存储员工请假信息，包括姓名、事由、起止日期。
- 考勤表：该表保存各个员工的出勤情况，是工资结算的重要依据。
- 库表：该表用于存储所有已建立档案的员工资料信息。主页表操作的数据都来源于该表。
- 参数表：该表保存系统用到的一些设置信息。

8.1.2　知识点一：名称

在 Excel 中应用单元格时，可以使用行列号，也可以首先对单元格进行命名，然后使用该名称调用该单元格。命名时，首先选中需要命名的单元格，然后依次选择【公式】|【定义名称】命令，弹出【新建名称】对话框，在【名称】文本框中输入新名称即可，如图 8-2 所示。



图 8-2　定义名称

通过 VBA 也可以建立名称，Names 对象代表了在单元区域上的定义名。Names 集合是应用程序或工作簿中所有 Name（名称）的集合。通过 VBA 添加名称时，可用 Add 方法创建名称并将其添加到集合中。下面是 Add 方法的一个实例：

```
Names.Add Name="新名称",RefersTo="=sheet1!$A$1:$G$10"
```

其中，Name 参数是该新名称的自定义名。RefersTo 参数必须以 A1 单元格样式表示法指定。上面的实例指定 sheet1 的 A1:G10 单元格区域的名称为“新名称”。

8.1.3　知识点二：使用 OnTime 方法

Application 对象的 OnTime 方法可以定时触发过程。定时可以是一个具体时间，也可以指定在某段时间之后。当在程序中需要运用到定时运行过程时，该方法十分方便。它的语法参照以下格式：

```
Application.OnTime(EarliestTime, Procedure, [LatestTime, Schedule])
```

其中用中括号括起来的是可选参数。EarliestTime 是过程运行的时间，Procedure 为过程名，LatestTime 是过程运行最迟时间。Schedule 参数是一个布尔值，当其为 False 时，清除前面定义的过程；为 True 时，将预定一个新过程，默认值为 True。下面是该方法使用的一个实例：

```
Application.OnTime Now + TimeValue("00:00:15"), "my_Procedure"
```

该语句的意义为从执行该语句开始后的 15 秒将自动运行 my_Procedure 过程。

8.1.4　知识点三：Range 对象的 Sort 方法

排序是在编程过程中经常碰到的事情，通过编程完成排序的工作也不算是件简单的事情，但是在 Excel 的编程中，要完成排序不一定需要自己编程完成该工作。Range 对象的 Sort 方法可以指定单元格区域进行排序工作，而且速度十分快捷。以下是该方法的格式：

```
Range.Sort([Key1, Order1,Key2,Type,Order2,Key3,Order3,Header,OrderCustom,MatchCase, Orientation,
SortMethod, DataOption1, DataOption2, DataOption3])
```

该方法的参数众多，但这些参数都是可选的。Key1、Order1、Key2、Type、Order2、Key3、

Order3 这些参数可以连续指定 3 个排序字段。Order 参数则指定对应字段的排序顺序。Header 参数确定第一行是否包含标题。OrderCustom 指定在自定义排序次序列表中的基于一的整数偏移。MatchCase 确定排序时是否区分大小写。Orientation 指定以升序还是降序排序。SortMethod 指定排序方法。DataOption 参数指定对应的字段区域中文本的排序方式。以下是该方法的一个应用实例：

```
Sheet1.Range("A1:F50").Sort Key1:=Range("C1"), _
    Order1:=xlAscending, Header:=xlGuess, OrderCustom:=1, MatchCase:=False, _
    Orientation:=xlTopToBottom, SortMethod:=xlPinYin, DataOption1:=xlSortNormal
```

以上代码将在 sheet1 表的 A1 到 F50 单元格区域内按照 C 列进行排序，排序方式为升序。

8.1.5 知识点四：CountIf 函数

CountIf 函数可以很容易地计算区域中满足条件的单元格个数。该函数是内置工作表函数，在工作表的单元格中输入该函数即可直接使用。当在 VBA 代码中使用时，必须在该函数之前采用 WorksheetFunction.CountIf 的方式调用。该函数的格式如下：

```
CountIf(Arg1, Arg2)
```

其中 Arg1 参数指定需要计算单元格个数的区域，Arg2 指定哪些单元格将被计算在内的条件，其形式可以为数字、表达式、单元格引用或文本。例如，条件可以表示为 32、"32"、">32"、"apples" 或 B4。可以在条件中使用通配符，包括问号 (?) 和星号 (*)。问号可匹配任意的单个字符；星号可匹配任意一串字符。如果要查找实际的问号或星号，则应在该字符前输入一个波形符 (~)。

在本章的实例的考勤表中使用到了该函数。例如，AH5 单元格的公式为 CountIf ($C5:$AG5,"√")。它统计的是 C5 到 AG5 单元格区域中包含“√”的单元格个数，即该员工正常出勤的天数。

8.1.6 知识点五：DateDiff 函数

DateDiff 函数可以计算两个日期间的时间间隔，最终返回的结果将根据参数指定的不同而不同。该函数的语法如下：

```
DateDiff(interval, date1, date2[, firstdayofweek[, firstweekofyear]])
```

interval 用来计算 date1 和 date2 的时间差的时间间隔，当指定为不同值时，返回的结果会不一样。date1、date2 分别指定两个日期，firstdayofweek 指定一个星期的第一天的常数。如果未予指定，则以星期日为第一天。firstweekofyear 指定一年的第一周的常数。如果未予指定，则以包含 1 月 1 日的星期为第一周。下面是该函数使用的一个实例：

```
Msgbox DateDiff("d", "2007-11-5", "2007-12-5")
```

以上代码将显示一个消息框，消息框中显示的数据为 30，即 2007 年 11 月 5 号与 2007 年

12 月 5 号的日期间隔为 30 天。如果将其中的“d”换成“m”，返回的结果为 1，因为两个日期的月份间隔为 1。

8.2　工作簿对象与表设计

员工档案管理系统包括 6 个表，分别是主页表、员工档案卡表、请假登记表、考勤表、库表和参数表。本小节将讲述这 6 个表的具体设计与工作簿对象过程。

8.2.1　主页表

下面详细讲述该主页表的设计方法与步骤。

（1）新建“主页”工作表。在 Excel 2007 中依次选择【开始】|【单元格】|【插入】|【插入工作表】命令，如图 8-3 所示。随后右击新插入工作表的标签，在弹出的快捷菜单中选择【重命名】命令，然后将其文字内容修改为“主页”。

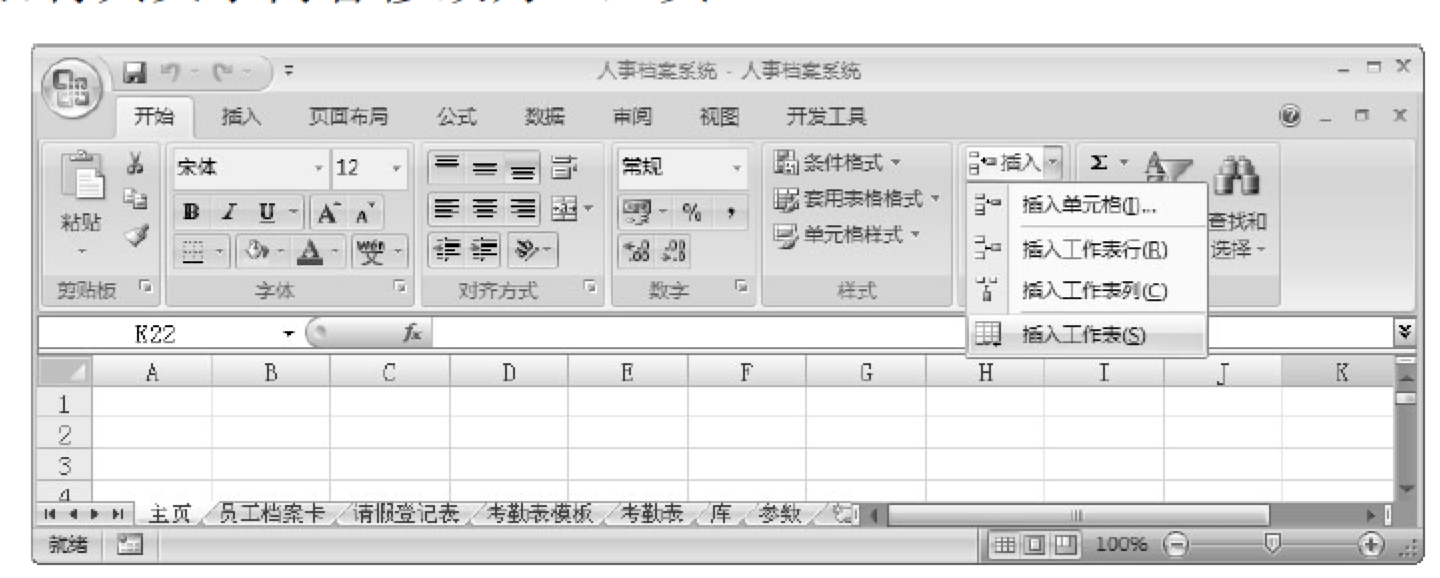


图 8-3　插入工作表操作示意图

（2）工作簿设置。选择 Excel 2007 左上角的 Office 菜单并单击【Excel 选项】按钮。在随后显示的【Excel 选项】对话框左侧选择【高级】选项，在其右侧找到工作簿与工作表显示选项设置，修改设置如图 8-4 所示。



图 8-4　【Excel 选项】设置

（3）添加外层矩形。在 Excel 2007 中依次选择【插入】|【形状】|【矩形】命令。在工作表空白区域单击鼠标左键并拖动以产生一个适当大小的矩形形状，随后右击矩形形状，在弹出的快捷菜单中选择【设置形状格式】命令，打开【设置形状格式】对话框。

（4）设置外层矩形格式。在【设置形状格式】对话框中选择【阴影】选项。随后在其右侧的【预设】下拉列表框中选择【外部】分类中的【右下斜偏移】选项。然后再次右击该矩形，在弹出的快捷菜单中依次选择【置于底层】|【置于底层】命令。最后再右击该矩形，在弹出的快捷菜单中选择【编辑文字】命令并输入文字内容为“员工管理系统”。

（5）添加按钮。在 Excel 2007 中依次选择【插入】|【形状】|【圆角矩形】命令。在工作表空白区域单击鼠标左键并拖动以产生一个适当大小的圆角矩形，右击该形状，在弹出的快捷菜单中选择【设置形状格式】命令，打开【设置形状格式】对话框。在【设置形状格式】对话框中选择【填充】选项并在其右侧选中【渐变填充】单选按钮，随后在【颜色】下拉列表框的【标准色】分类中选择【橙色】选项。然后再设置该形状的阴影效果，其设置方法与步骤（4）相同，这里不再多做说明。最后右击该矩形，在弹出的快捷菜单中选择【编辑文字】命令后输入文字内容为“员工资料管理”。

（6）右击刚创建圆角矩形并选择【复制】命令，将其粘贴两次。将两圆角矩形拷贝的文字内容分别修改为“考勤签到”和“请假登记”。其修改的方法在步骤（4）和步骤（5）中都有说明，这里不再赘述。

（7）为形状按钮指定宏。右击各个形状按钮，在弹出的快捷菜单中选择【指定宏】命令，打开【指定宏】对话框。在【指定宏】对话框中分别指定各个按钮的宏过程。其相应的宏过程名称与其文字内容相同。

设计好的主页表界面如图 8-5 所示。



图 8-5　员工管理系统主页

8.2.2　员工档案卡表界面设计

员工档案卡表界面设计步骤如下：

（1）修改工作表标签名。右击工作表标签，在弹出的快捷菜单中选择【重命名】命令，随后输入文字内容为“员工档案卡”。

（2）合并单元格。工作表中很多地方需要合并单元格，以便于对齐文字及输入。例如

D5:F5 单元格区域对应“所学专业”的内容。合并时首先选中该区域并右击鼠标，在弹出的快捷菜单中选择【设置单元格格式】命令，打开【设置单元格格式】对话框，选择【对齐】选项卡。随后在【文本控制】分类中选中【合并单元格】复选框。

（3）设置边框格式。选中 A2:G19 单元格区域并右击鼠标，在随后弹出的快捷菜单中选择【设置单元格格式】命令，打开【设置单元格格式】对话框，选择【边框】选项卡，在【预置】分类中选择【外边框】和【内部】复选框即可。

（4）设计标题。在合并后的单元格区域 A1:G1 中双击并输入文字内容为“职工档案卡”。

（5）文字部分。需要预先设计的文字部分是对应项目的提示文字，即在库表中的列标题。主页表的文字部分是通过链接库表对应的列标题实现的。例如“职工编号”提示文字，其链接的是库表中的 A1 单元格，这里使用公式“=库!A1”即可。其他文字部分与此类似。关于为何要使用链接，将在查询功能模块详细讲述。

（6）插入图像控件。通过图像控件，可以在档案卡中显示员工照片。在 Excel 2007 中依次选择【开发工具】|【控件】|【插入】命令。然后在 ActiveX 控件分类中选择图像控件。随后在员工档案卡表中插入一图像控件并将其位置与大小对齐到 G2:G6 单元格区域。

（7）设置图像控件显示效果。图像控件默认的图像显示时会将超出部分自动剪裁。为了防止部分员工图片过大而造成只能显示一部分的情况出现，应该设置图像控件的显示模式。在图像控件上右击，在弹出的快捷菜单中选择【属性】命令，打开【属性】对话框，将 PictureSizeMode 属性修改为“3-fmPictureSizeModeZoom”即可。该表的界面如图 8-6 所示。

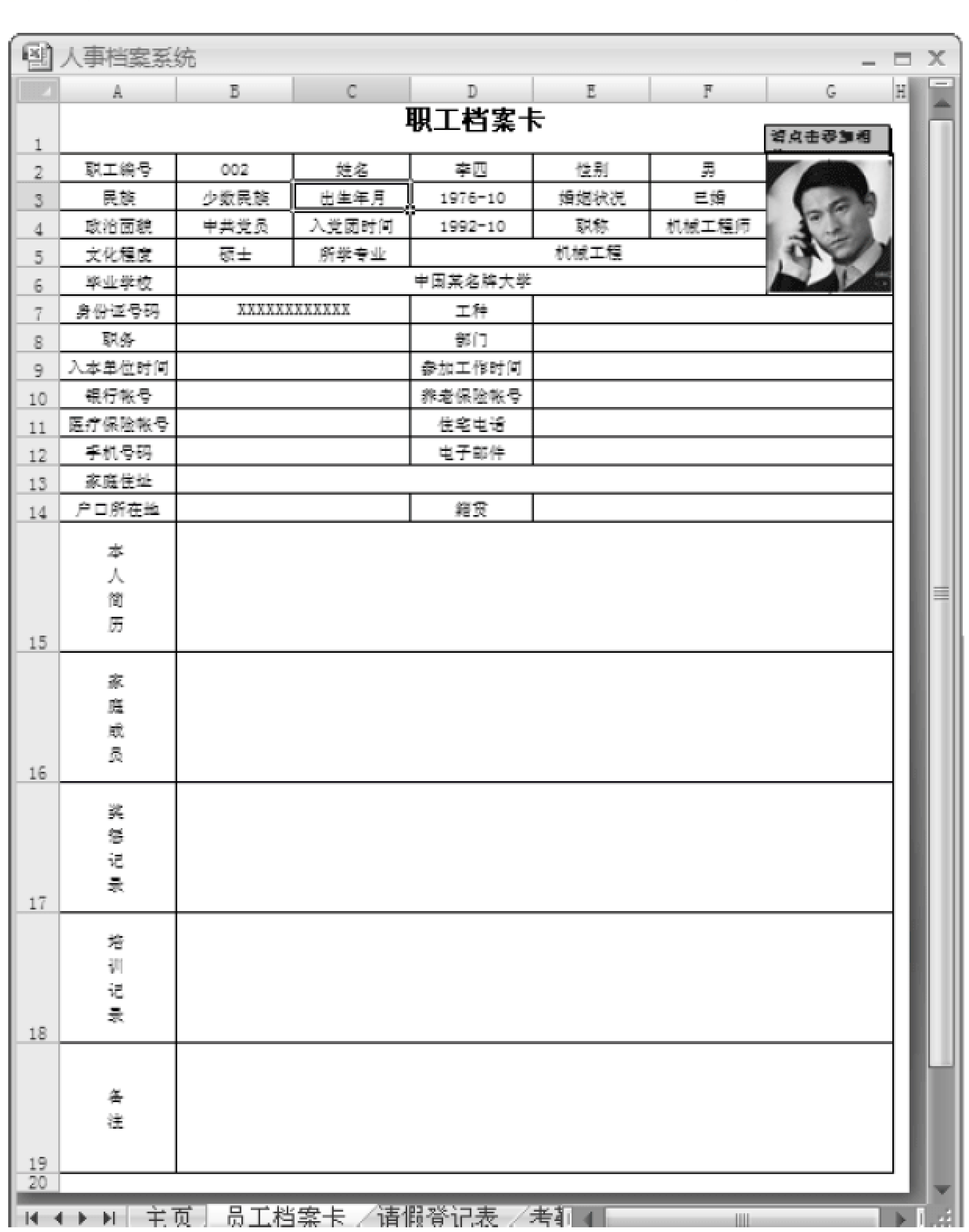


图 8-6　员工档案卡表

8.2.3 员工档案卡表代码设计

在员工档案卡表中包括 5 个事件过程，分别是图像控件上鼠标滑动事件过程、图像控件单击事件过程、工作表改变事件过程、工作表激活事件过程以及工作表失去激活事件过程。下面分别讲述 5 个事件过程的作用与代码。

1．图像控件上鼠标滑动事件过程

当鼠标在图像控件上滑动时，需要将已经在 G2 单元格中设置好的批注显示出来，当该批注显示了 1 秒后应该被自动隐藏起来。该事件过程的代码如下：

```
Private Sub Image1_MouseMove(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single,
ByVal Y As Single)
    '显示 G2 批注提示
    Range("g2").Comment.Visible = True
    '将批注提示显示 1 秒后隐藏
    Application.OnTime Now + TimeValue("00:00:1"), "Hidden_Comment"
End Sub
```

2．图像控件单击事件过程

当单击图像控件时，弹出【文件获取】对话框，以获取员工照片的存储位置。该图像的位置将会被保存在 G2 单元格中。

```
Private Sub Image1_Click() '单击图像控件添加相片地址
   If Range("B2").HasFormula = True Then End
   Dim Fd As FileDialog
   Set Fd = Application.FileDialog(msoFileDialogFilePicker)
   With Fd
          .Title = "选取个人相片"
          .Filters.Add "Images", "*.gif; *.jpg; *.jpeg", 1 '图像类型
          .AllowMultiSelect = False   '不能多选
          If .Show = -1 Then
              ActiveSheet.Unprotect
              Range("g2").Value = Fd.SelectedItems(1)
              ActiveSheet.Protect DrawingObjects:=True, Contents:=True, Scenarios:=True
              ActiveSheet.EnableSelection = xlUnlockedCells
              Picture_Load '显示图片
          End If
   End With
   Set Fd = Nothing
   Range("c3").Select
End Sub
```

3．工作表改变事件过程

当工作表内容发生变化时，需要检测当前是否处于查询状态。当处于查询状态时，需要给查询过程传递参数，并执行该查询过程。

```
Private Sub Worksheet_Change(ByVal Target As Range) '记录寻找
If Find_Status = True Then
    Data_Search Target.Offset(0, -1).Formula, Target.Value
End If
End Sub
```

4．工作表激活事件过程

当工作表被激活时，系统将为该人事档案卡表生成一个菜单栏。该菜单栏包含了所有对人事档案卡表的操作，该菜单栏的界面如图 8-7 所示。该工作表激活事件过程的流程图如图 8-8 所示。

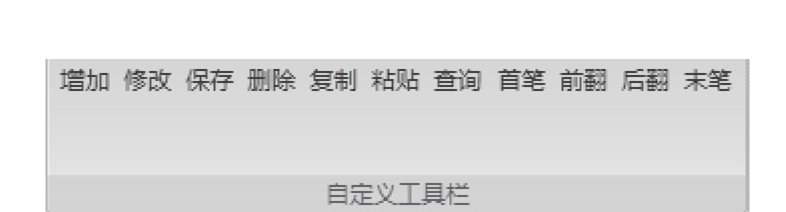


图 8-7　人事档案卡表菜单

添加人事档案菜单栏
↓
在人事档案菜单栏中添加各个按钮并设置各个按钮的执行过程
↓
将员工档案卡显示记录定位到库表最后一条
↓
调用图片加载过程加载最后一个员工的图片

图 8-8　工作表激活事件过程流程图

生成该菜单栏的代码如下：

```
Private Sub Worksheet_Activate()
    Application.ScreenUpdating = False
    With Application.CommandBars.Add("人事档案菜单")
        .Visible = True
        .Position = msoBarFloating
        Dim myButton As CommandBarButton
        With .Controls
            Set myButton = Application.CommandBars("人事档案菜单").Controls.Add
            myButton.Caption = "增加"
            myButton.Style = msoButtonCaption
            myButton.BeginGroup = True
            myButton.OnAction = "data_add"
            myButton.TooltipText = "增加记录"
            myButton.Enabled = True
        End With
        With .Controls
            Set myButton = Application.CommandBars("人事档案菜单").Controls.Add
            myButton.Caption = "修改"
            myButton.Style = msoButtonCaption
            myButton.BeginGroup = True
            myButton.OnAction = "data_modify"
            myButton.TooltipText = "修改记录"
            myButton.Enabled = True
        End With
        With .Controls
```

```
            Set myButton = Application.CommandBars("人事档案菜单").Controls.Add
            myButton.Caption = "保存"
            myButton.Style = msoButtonCaption
            myButton.BeginGroup = True
            myButton.OnAction = "data_save"
            myButton.TooltipText = "保存记录"
            myButton.Enabled = True
        End With
        With .Controls
            Set myButton = Application.CommandBars("人事档案菜单").Controls.Add
            myButton.Caption = "删除"
            myButton.Style = msoButtonCaption
            myButton.BeginGroup = True
            myButton.OnAction = "data_del"
            myButton.TooltipText = "删除记录"
            myButton.Enabled = True
        End With
        With .Controls
            Set myButton = Application.CommandBars("人事档案菜单").Controls.Add
            myButton.Caption = "复制"
            myButton.Style = msoButtonCaption
            myButton.BeginGroup = True
            myButton.OnAction = "data_copy"
            myButton.TooltipText = "复制记录"
            myButton.Enabled = True
        End With
        With .Controls
            Set myButton = Application.CommandBars("人事档案菜单").Controls.Add
            myButton.Caption = "粘贴"
            myButton.Style = msoButtonCaption
            myButton.BeginGroup = True
            myButton.OnAction = "data_paste"
            myButton.TooltipText = "粘贴记录"
            myButton.Enabled = True
        End With
        With .Controls
            Set myButton = Application.CommandBars("人事档案菜单").Controls.Add
            myButton.Caption = "查询"
            myButton.Style = msoButtonCaption
            myButton.BeginGroup = True
            myButton.OnAction = "DataFind_Status"
            myButton.TooltipText = "查询记录"
            myButton.Enabled = True
        End With
        With .Controls
            Set myButton = Application.CommandBars("人事档案菜单").Controls.Add
            myButton.Caption = "首笔"
            myButton.Style = msoButtonCaption
            myButton.BeginGroup = True
```

```
            myButton.OnAction = "data_first"
            myButton.TooltipText = "首笔记录"
            myButton.Enabled = True
        End With
        With .Controls
            Set myButton = Application.CommandBars("人事档案菜单").Controls.Add
            myButton.Caption = "前翻"
            myButton.Style = msoButtonCaption
            myButton.BeginGroup = True
            myButton.OnAction = "data_pageup"
            myButton.TooltipText = "前翻记录"
            myButton.Enabled = True
        End With
        With .Controls
            Set myButton = Application.CommandBars("人事档案菜单").Controls.Add
            myButton.Caption = "后翻"
            myButton.Style = msoButtonCaption
            myButton.BeginGroup = True
            myButton.OnAction = "data_pagedown"
            myButton.TooltipText = "后翻记录"
            myButton.Enabled = True
        End With
        With .Controls
            Set myButton = Application.CommandBars("人事档案菜单").Controls.Add
            myButton.Caption = "末笔"
            myButton.Style = msoButtonCaption
            myButton.BeginGroup = True
            myButton.OnAction = "data_last"
            myButton.TooltipText = "末笔记录"
            myButton.Enabled = True
        End With
    End With
    Worksheets("参数").Range("a2").Value = Worksheets("库").UsedRange.Rows.Count -1
    '获得末笔记录
    Application.ScreenUpdating = True
    Picture_Load
End Sub
```

5. 工作表失去激活事件过程

当通过不同的方式从该员工档案卡表切换到其他工作表时，将发生工作表失去激活事件。此时需要删除在工作表激活事件中生成的菜单栏，因为该菜单栏只支持员工档案卡表中的操作。该事件过程的代码如下：

```
Private Sub Worksheet_Deactivate()
    Application.CommandBars("人事档案菜单").Delete
End Sub
```

8.2.4 请假登记表设计

请假登记表主要记录请假员工的姓名、请假事由以及请假的期限。该表的结构十分简单，结构如图 8-9 所示。

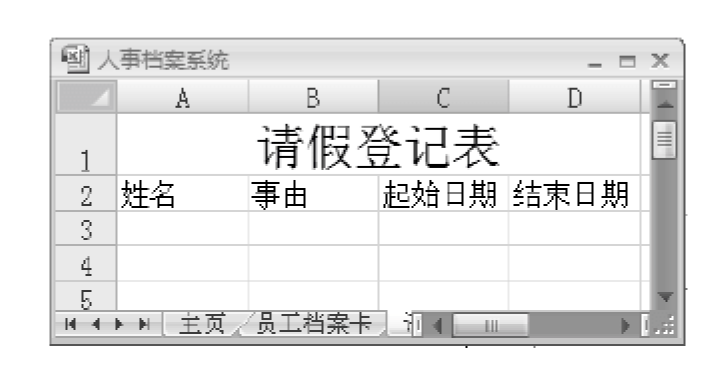


图 8-9　请假登记表界面

8.2.5 考勤表设计

考勤表用于登记员工一个月内的考勤情况，该表对于财务进行工资报酬计算十分重要。该表的结构如图 8-10 所示。



图 8-10　考勤表界面

8.2.6 库表设计

库表保存了所有登记了的员工的信息。库表的界面如图 8-11 所示，以表列的形式存储了所有的数据。该表的设计十分简单，此处不再列出设计步骤。

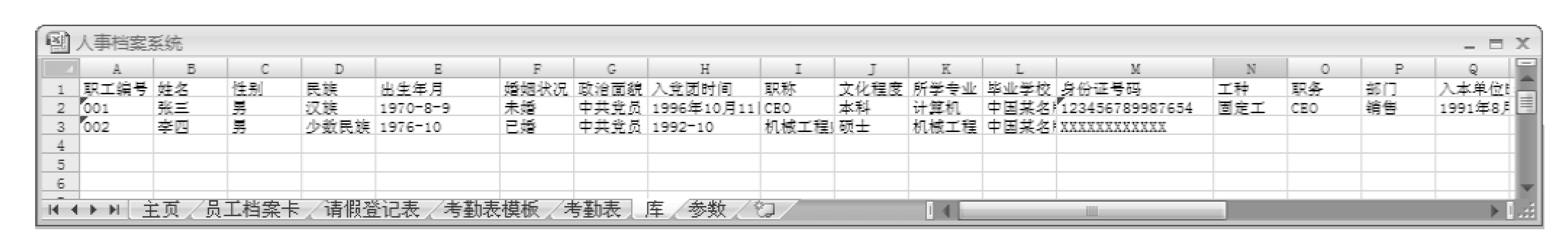


图 8-11　库表界面

8.2.7 参数表设计

参数表中存储了在该系统中需要用到的参数，对应的参数都附加相关的说明，如图 8-12 所示。

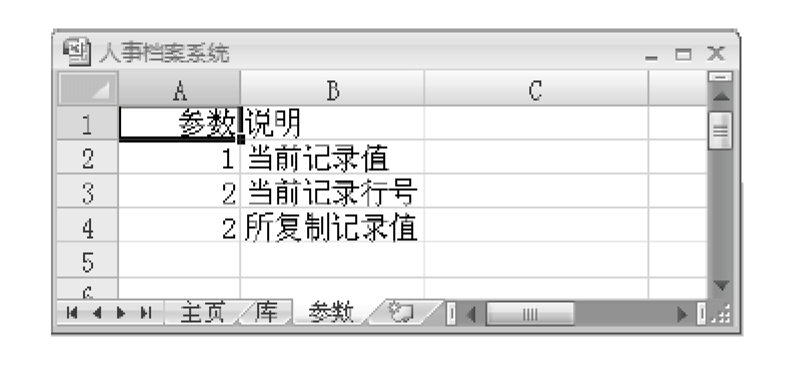


图 8-12　参数表设计

8.2.8 工作簿对象设计

工作簿对象包含了 3 个事件过程，分别是工作簿开启事件过程、工作簿关闭事件过程和工作簿保存事件过程。工作簿的开启事件过程用于初始化工作簿的一些状态。工作簿关闭事

件过程用于删除系统建立的菜单并且保存工作簿。工作簿保存事件过程用于保存员工库表记录数以及清空图像控件的显示。详细代码解释如下：

```
Private Sub Workbook_BeforeClose(Cancel As Boolean)
    Workbooks("人事档案.xls").Save
    Application.Caption = "Microsoft Excel"
End Sub

Private Sub Workbook_BeforeSave(ByVal SaveAsUI As Boolean, Cancel As Boolean)
    Worksheets("参数").Range("a2").Value = Worksheets("库").UsedRange.Rows.Count '获得末笔记录
    Sheets("员工档案卡").Image1.Picture = LoadPicture("")
     '此命令行作用为设置图像控件相片为空,否则保存会占用较多空间
End Sub

Private Sub Workbook_Open()
    Application.Caption = "人事档案系统"
    Application.MoveAfterReturnDirection = xlToRight
    ActiveWindow.Caption = "人事档案系统"
End Sub
```

8.3　设计员工档案卡模块代码

员工的个人档案资料都是在员工档案卡中进行管理的。当该表被激活时，会生成对应的自定义工具栏，该工具栏中包含了所有在该工作表中需要完成操作的命令按钮，包括记录增加、修改、删除、查询、浏览以及返回主页的【返回】按钮。下面介绍该员工档案模块的详细代码。

8.3.1　变量定义

在该模块中有部分自定义变量。在介绍该部分代码前，首先应弄清楚这些变量的意义，理解相应代码的意图。以下是变量定义解释：

```
'该单元格对象用在 For...Each 循环中，用于指代单元格区域中单个的单元格
Dim Input_Cell As Range
'以下变量分别代表当前行号，当前列号，库表中末笔记录行号
Dim Current_Row, Current_Col, Large_Row As Integer
'用于界定当前是否处于查询状态
Public Find_Status As Boolean
```

8.3.2　记录新增操作

当添加员工档案时，在系统中需要完成部分初始工作。这些工作包括设置非查询状态、检查是否已处于新增记录操作、解除工作表保护。其增加步骤如图 8-13 所示。

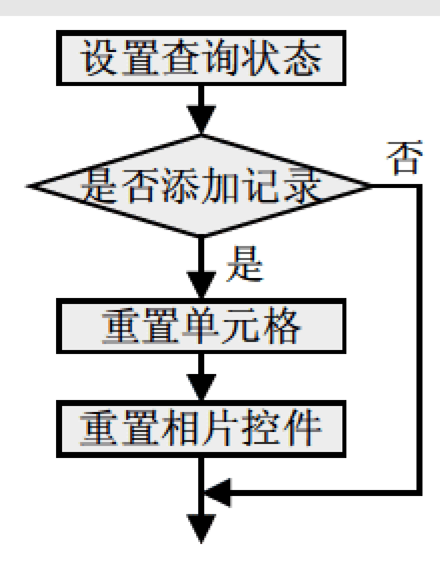


图 8-13　记录增加步骤

```
'增加记录
Sub Data_Add()
    '设置当前为非查询状态
    Find_Status = False
    '检查职工编号单元格是否已经有记录，当该单元格为空时，说明当前正完成增加记录操作，退出该过程
    If Range("B2").HasFormula = False Then End
    Application.ScreenUpdating = False                      '关闭应用程序刷新
    ActiveSheet.Unprotect                                   '解除工作表锁定
    Large_Row = Worksheets("库").UsedRange.Rows.Count        '获得末笔记录行号
    Worksheets("参数").Range("a2").Value = Large_Row          '保存末笔记录行号
    For Each Input_Cell In Range("Data_Area")               '把记录区域置空
        Input_Cell.NumberFormatLocal = "@"                  '将当前单元格数据格式设置为常用
        Input_Cell.Value = ""                               '设置当前单元格值为空
    Next
    Sheets("员工档案卡").Image1.Picture = LoadPicture("")     '清空相片控件
    Sheet_Unlock                                            '设定输入状态
    Application.ScreenUpdating = True                       '开启应用程序刷新
End Sub
```

8.3.3 记录修改操作

当修改记录数据时，需要完成一些初始工作，包括设置当前查询状态、解锁工作表保护、重置单元格格式以及将单元格的公式转换为实际值等。修改记录操作的操作步骤如图 8-14 所示。以下是该过程的详细代码解释：

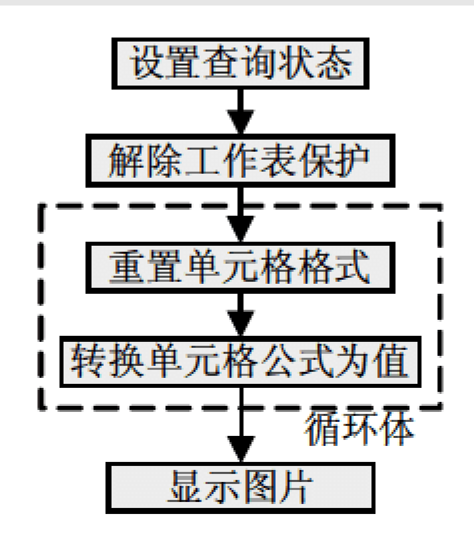


图 8-14 修改记录操作步骤

```
'修改记录
Sub Data_Modify()
    '设置当前为非查询状态
    Find_Status = False
    Application.ScreenUpdating = False              '关闭工作表刷新
    ActiveSheet.Unprotect                           '解除工作表保护
    '重置记录单元格格式与数据
    For Each Input_Cell In Range("Data_Area")
        Input_Cell.NumberFormatLocal = "@"          '设置单元格格式
        Input_Cell.Value = Input_Cell.Value         '去公式，置值
    Next
    Sheet_Unlock
    Application.ScreenUpdating = True               '关闭工作表刷新
    Picture_Load '显示图片
End Sub
```

8.3.4 记录删除操作

进行记录删除操作前，需要确认当前为非查询状态、检查当前是否有浏览记录。在进行删除操作前，根据库表中的记录行数确认是否有可能完成删除操作。如图 8-15 所示的是该过程的执行流程。

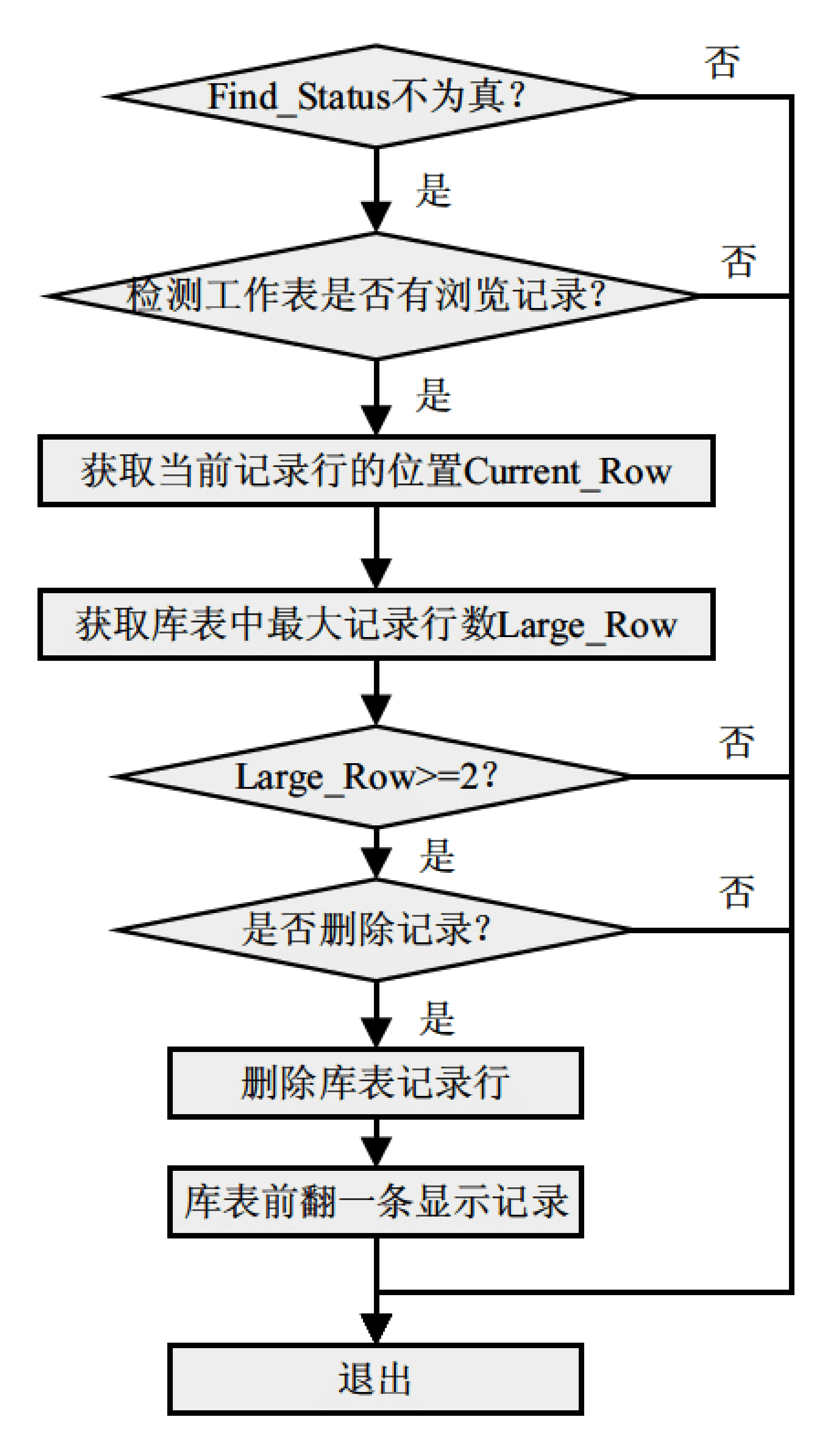


图 8-15　删除记录过程流程

以下是该过程的详细代码解释：

```
'删除记录
Sub Data_Del()
Dim Yn As Integer
    If Find_Status = True Then End                         '确认当前为非查询状态
    If Range("B2").HasFormula = False Then End             '检查当前是否有浏览记录
    Current_Row = Worksheets("参数") .Range("a3").Value    '当前记录行的位置
    Large_Row = Worksheets("库").UsedRange.Rows.Count      '库表中最大记录行数
    If Large_Row >= 2 Then                                 '判断记录是否为空
        '提示是否删除记录
        Yn = MsgBox("确定删除当前记录吗", vbOKCancel, "删除记录")
        If Yn = 1 Then
            '将库表中对应的记录行删除
            Worksheets("库").Rows(Current_Row & ":" & Current_Row).Delete Shift:=xlUp
            '将员工档案卡显示的记录指向上一条
            Data_PageUp                                    '前翻一条记录
        End If
    Else
        MsgBox "对不起,没有记录可以删除"
        End
    End If
End Sub
```

8.3.5 记录保存操作

单击【保存】按钮后，同前面的按钮一样，也会首先检测当前的状态，以确定是否需要完成保存操作，然后把主页工作表的数据保存到资料工作表中。该过程的流程图如图 8-16 所示。

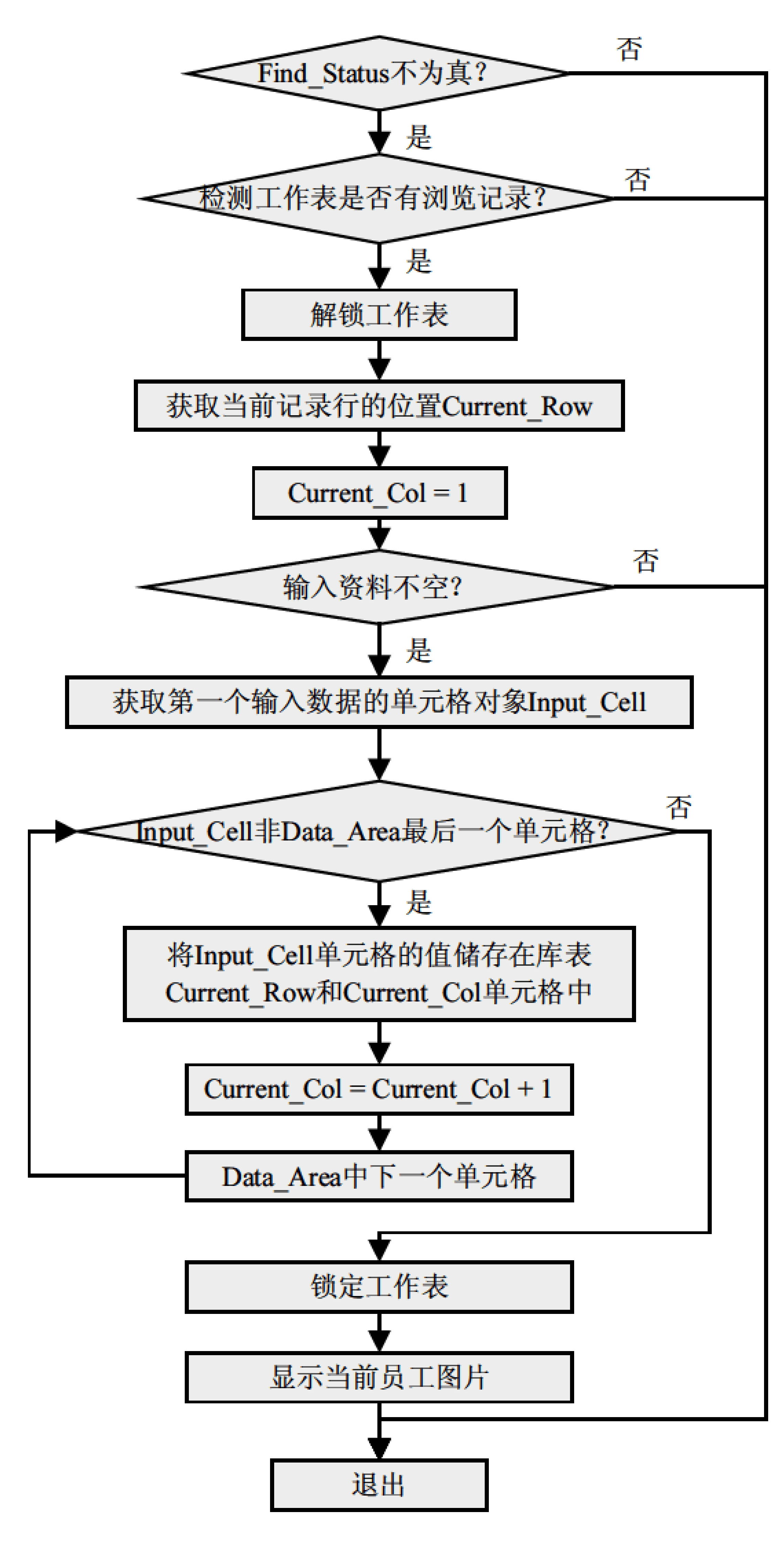


图 8-16 记录保存过程流程图

以下是该过程的详细代码解释：

```
'保存记录
Sub Data_Save()
    If Find_Status = True Then End
    If Range("B2").HasFormula = True Then End
    Application.ScreenUpdating = False
```

```
        ActiveSheet.Unprotect
        Current_Row = Worksheets("参数").Range("a3").Value   '获得当前记录行号
        Current_Col = 1
        '判断输入资料是否为空
        If Application.WorksheetFunction.CountA(Range("Data_Area")) > 2 Then
            '循环，把每一个数据保存到资料工作表中
            For Each Input_Cell In Range("Data_Area")
                With Worksheets("库")
                    .Cells(Current_Row, Current_Col).Value = Input_Cell.Value
                End With
                Input_Cell.NumberFormatLocal = "G/通用格式"
                Input_Cell.FormulaR1C1 = "=OFFSET(库!R1C" & Current_Col & ",参数!R2C1,0,,)"
                Current_Col = Current_Col + 1
            Next
            Sheet_Lock
            Picture_Load '显示图片
        Else
            Sheet_Unlock
        End If
        Application.ScreenUpdating = True
End Sub
```

8.3.6　记录复制/粘贴操作

记录的复制与粘贴操作是为了方便记录的新增而设的。有时候新员工的部分资料可能与已建立资料的某员工资料一致，此时可以使用复制/粘贴操作快速建立员工资料。复制记录按钮过程十分简单，只需要保存当前显示员工的记录号即可。粘贴时直接使用该保存值确定需要复制的记录所在行号。以下是这些事件代码的详细解释：

```
Sub Data_Copy() '复制记录
        If Find_Status = True Then End
        '将当前显示员工的行号记录到参数表中
        With Worksheets("参数")
            .Range("a4").Value = .Range("a3").Value              '保存当前显示员工所在行号
        End With
End Sub

Sub Data_Paste() '粘贴记录
Dim Yn As Integer
        If Find_Status = True Then End
        If Range("B2").HasFormula = True Then End
        Application.ScreenUpdating = False
        Current_Row = Worksheets("参数").Range("a4").Value     '获得当前记录行号
        Current_Col = 1
        '判断输入资料是否为空
        If Application.WorksheetFunction.CountA(Range("Data_Area")) > 2 Then
            '当前显示有记录时，提示是否覆盖数据
            Yn = MsgBox("当前记录已有数据存在,覆盖吗", vbOKCancel, "粘贴记录")
            '当用户确认覆盖数据时，从库表中获取数据覆盖到表中
            If Yn = 1 Then
```

```
            ActiveSheet.Unprotect
            For Each Input_Cell In Range("Data_Area")
                Input_Cell.Value = Worksheets("库").Cells(Current_Row, Current_Col).Value
                Current_Col = Current_Col + 1
            Next
            Sheet_Unlock
            Picture_Load                                              '显示图片
        End If
    Else
        ActiveSheet.Unprotect
        For Each Input_Cell In Range("Data_Area")
            Input_Cell.Value = Worksheets("库").Cells(Current_Row, Current_Col).Value
            Current_Col = Current_Col + 1
        Next
        Sheet_Unlock
        Picture_Load '显示图片
    End If
    Application.ScreenUpdating = True
End Sub
```

8.3.7 Sheet_Formula 过程

Sheet_Formula 过程用于给设计员工档案表 Data_Area 中各个单元格赋予公式。这些公式可以从库表中获取用户浏览员工的信息。该公式通过一个 Offset 函数获取这些信息。该函数以库表第一行、第 Current_Col 列为基准，向下偏移一个指定值。该指定值随着用户单击【浏览】按钮而发生变化。公式也会根据该值使员工档案卡当前显示的员工记录发生变化。该过程的流程图如图 8-17 所示。

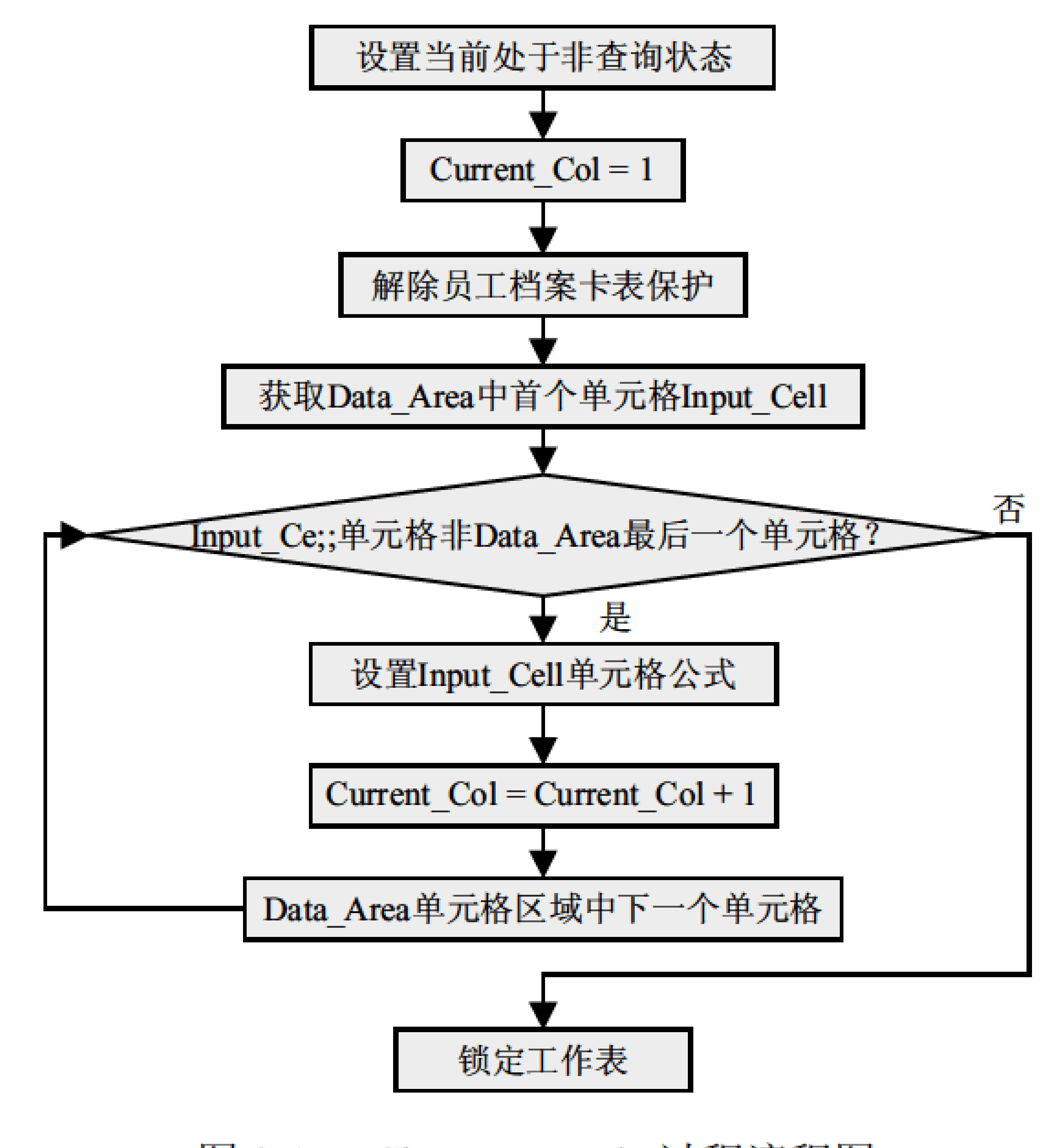


图 8-17　Sheet_Formula 过程流程图

以下是该过程的详细代码解释：

```
'输入区域赋予公式
Sub Sheet_Formula()
      Find_Status = False
      Application.ScreenUpdating = False
      Current_Col = 1
      ActiveSheet.Unprotect
      For Each Input_Cell In Range("Data_Area")
            Input_Cell.NumberFormatLocal = "G/通用格式"
            '将库表中的数据通过引用到员工资料中
            'Input_Cell 单元格获取的值为以库表第一行、第 Current_Col 列为基准，向下偏移参数表中 A2
单元格值的行后获得的那个单元格的值
            Input_Cell.FormulaR1C1 = "=OFFSET(库!R1C" & Current_Col & ",参数!R2C1,0,,)"
            '将列偏移量增加 1
            Current_Col = Current_Col + 1
      Next
      Sheet_Lock
      Application.ScreenUpdating = True
End Sub
```

8.3.8　记录浏览操作

浏览记录的功能是通过 4 个按钮共同完成的，这些按钮包括首笔、前翻、后翻和末笔。4 个按钮的功能分别通过各自的过程代码完成，其中首笔与末笔过程的流程、前翻与后翻过程的流程相差不大，因而后面加以介绍时，只给出首笔和前翻的流程图。以下是这 4 个按钮的功能以及实现过程描述：

1．首笔按钮过程

当用户在员工档案卡工作表中单击【首笔】按钮后，表中显示的记录将更换为第一个员工的信息。如果工作表中所有显示员工信息的单元格内具有链接公式，此时只需要修改参数表 A2 单元格中存储的单元格偏移量即可以更新工作表中所有的员工信息。如果没有链接公式，此时通过调用 Sheet_Formula 过程向工作表中写入公式。该过程的流程图如图 8-18 所示。

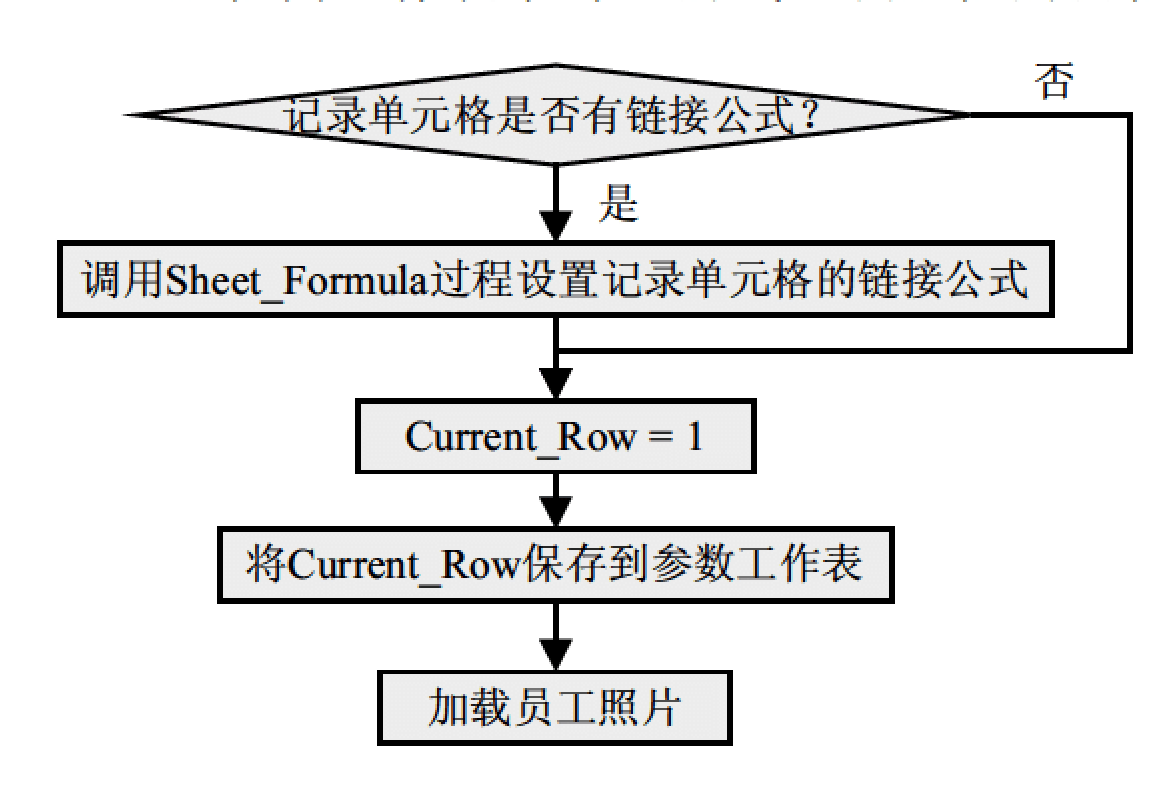


图 8-18　【首笔】按钮单击事件过程流程图

2．前翻按钮过程

【前翻】按钮被单击时，员工档案卡工作表显示的员工记录将移动到上一条。首先程序检测工作表中显示信息单元格是否有链接公式，然后获取先前显示记录的行偏移数并根据该变量的大小决定是否进行前翻操作。当该值大于或等于 2 时，程序将允许前翻，并把行偏移值减去 1 保存该值。最后，程序调用 Picture_Load 过程将前翻后的员工的图片显示出来。该过程的流程图如图 8-19 所示。

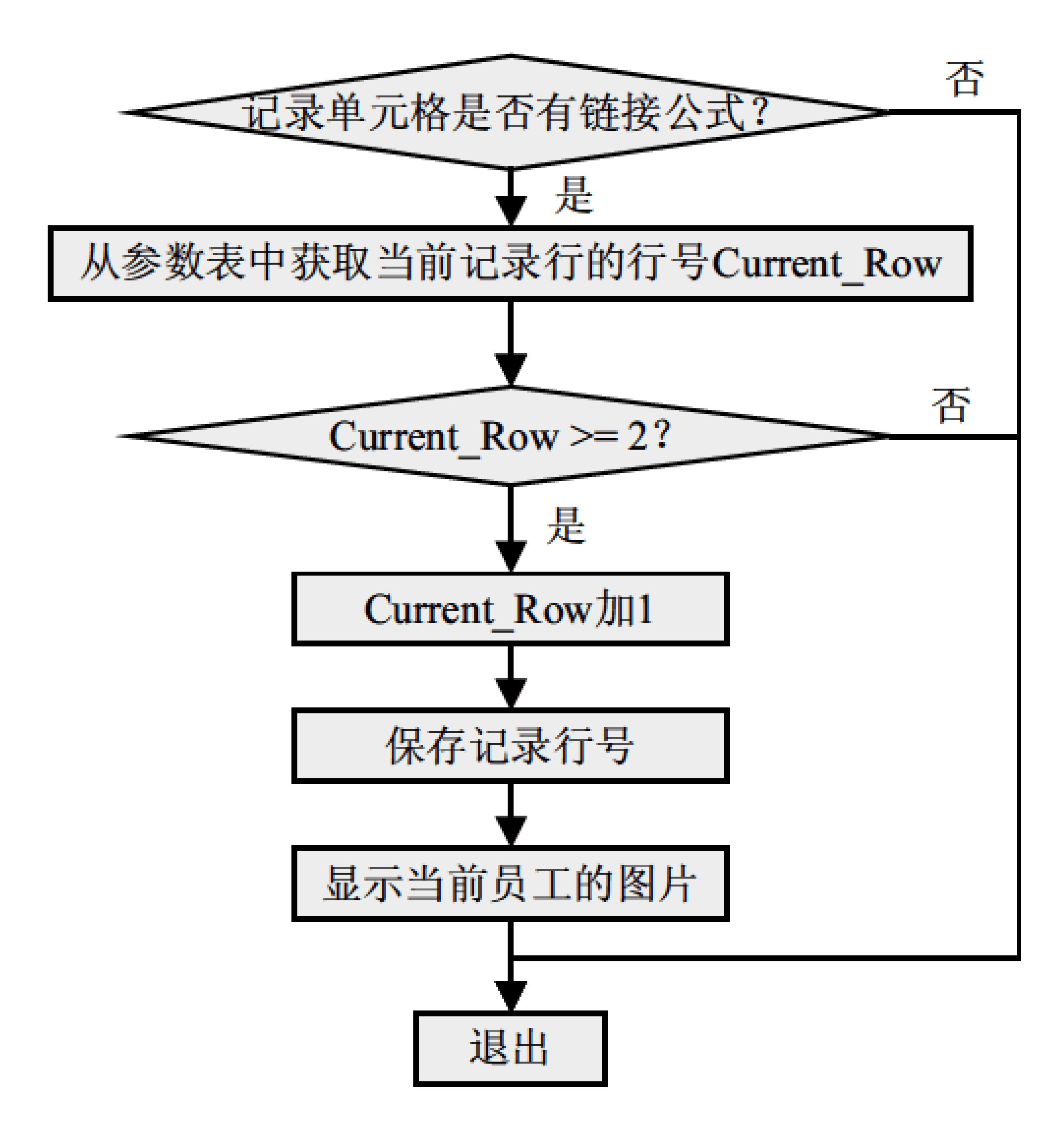


图 8-19 【前翻】按钮单击事件过程流程图

3．后翻按钮过程

【后翻】按钮被单击时，员工档案卡工作表显示的员工记录将移动到下一条。

首先程序检测工作表中显示信息单元格是否有链接公式，然后获取先前显示记录的行偏移数以及库表员工记录的最大偏移量。根据这两个变量的大小关系决定是否进行后翻操作。当行偏移值小于等于最大偏移时，程序将允许后翻，并把行偏移值加上 1 并保存该值。最后，程序调用 Picture_Load 过程将后翻后的员工的图片显示出来。

4．末笔按钮过程

当用户在员工档案卡工作表中单击【末笔】按钮后，表中显示的记录将更换为末条员工的信息。如果工作表中所有显示员工信息的单元格内具有链接公式，此时只需要修改参数表 A2 单元格中存储的单元格偏移量即可以更新工作表中所有的员工信息。如果没有链接公式，此时通过调用 Sheet_Formula 过程向工作表中写入公式。

以下是这些过程的详细代码解释：

```
'首笔记录
Sub Data_First()
    '如果单元格不包含公式，执行 Sheet_Formula 转换公式过程
    If Range("B2").HasFormula = False Then Sheet_Formula
    '制定当前行为 1，并且将该值保存到参数表中
```

```
    Current_Row = 1
    Worksheets("参数").Range("a2").Value = Current_Row
    Picture_Load                                               '显示图片
End Sub

'向前翻
Sub Data_PageUp()
    '如果单元格不包含公式，执行 Sheet_Formula 转换公式过程
    If Range("B2").HasFormula = False Then Sheet_Formula
    '从参数表中获取当前记录行的行号
    Current_Row = Worksheets("参数").Range("a2").Value
    '当当前记录行的行号大于等于 2 时，向前翻动记录才具有意义
    If Current_Row >= 2 Then
        '将当前记录行的行号指向即将显示记录的行号
        Current_Row = Current_Row - 1
        '保存记录行号
        Worksheets("参数").Range("a2").Value = Current_Row
        Picture_Load '显示图片
    Else
        MsgBox "已到最前面的记录了!"
    End If
End Sub

'向后翻
Sub Data_PageDown()
    '如果单元格不包含公式，执行 Sheet_Formula 转换公式过程
    If Range("B2").HasFormula = False Then Sheet_Formula
    '从参数表中获取当前行的行号
    Current_Row = Worksheets("参数").Range("a2").Value
    '计算员工记录的最大行
    Large_Row = Worksheets("库").UsedRange.Rows.Count - 1
    '对比 Current_Row 与 Large_Row 确定是否已经位于末笔记录
    If Current_Row < Large_Row Then
        '当没有到末笔记录时，将 Current_Row 累加 1，并将该值保存在参数表中
        Current_Row = Current_Row + 1
        Worksheets("参数").Range("a2").Value = Current_Row
        Picture_Load '显示图片
    Else
        MsgBox "已到最后面的记录了!"
    End If
End Sub

'末笔记录
Sub Data_Last()
    '如果单元格不包含公式，执行 Sheet_Formula 转换公式过程
    If Range("B2").HasFormula = False Then Sheet_Formula
    Current_Row = Worksheets("库").UsedRange.Rows.Count -1        '获得末笔记录
    Worksheets("参数").Range("a2").Value = Current_Row
    Picture_Load                                                   '显示图片
End Sub
```

8.3.9 记录的查询操作

记录的查询工作分为 3 个步骤：首先是单击【查询】按钮时，初始化查询状态；然后通过工作表改变事件激发查询过程；最后通过查询过程获取查询结果，将查询结果显示到员工档案卡表中。工作表改变事件过程参见员工档案卡表代码设计一节。本小节只包含了整个查询过程的中首尾两个过程代码。以下是这两个过程功能的说明：

1．查询初始化过程

当单击【查询】按钮后，系统需要完成众多的初始化操作，以便于程序获取查询条件并存储。程序需要解锁工作表、置空输入单元格、清空 Image 控件显示以及设置 Find_Status 变量为真。Find_Status 变量用于标记当前正处于查询的员工记录状态，以区别于浏览和编辑。如图 8-20 所示的是该过程的流程图。

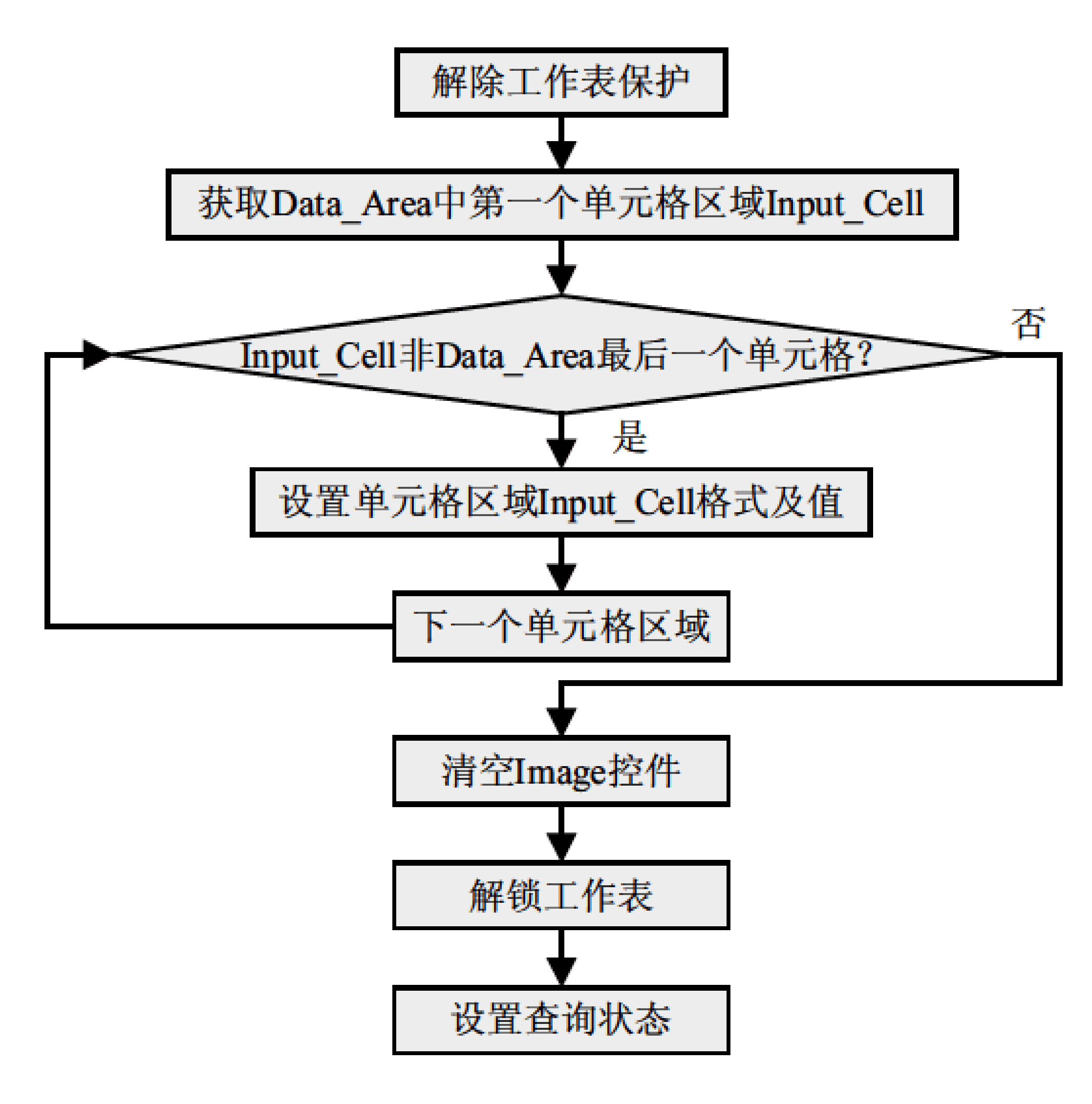


图 8-20　查询初始化过程流程图

2．查询记录过程

该过程接受两个参数：一个是查询信息在库表所在列的标题单元格地址 myAddress，另一个是查询的值 Find_Value。在员工档案卡工作表中，每个输入信息单元格左侧的提示单元格的值都是链接到库表的，因而要获取 myAddress，只需要读取查询条件输入单元格左侧单元格的公式即可。

由于查询过程的流程比较复杂，这里将详细介绍该过程的流程，程序首先根据 myAddress 参数获取查询列列标题单元格对象，然后对该列进行排序，以加快查询速度。接着通过 Find 方法在该列中找到等于该值的单元格。当找到该单元格时，记忆下该单元格的行号，并通过 Sheet_Formula 过程将查询到的员工信息显示出来并显示该员工的图片，最后标记处于非查询状态。当未找到记录时，提示未找到记录并标记当前处于查询状态。如图 8-21 所示的是查询记录过程流程。

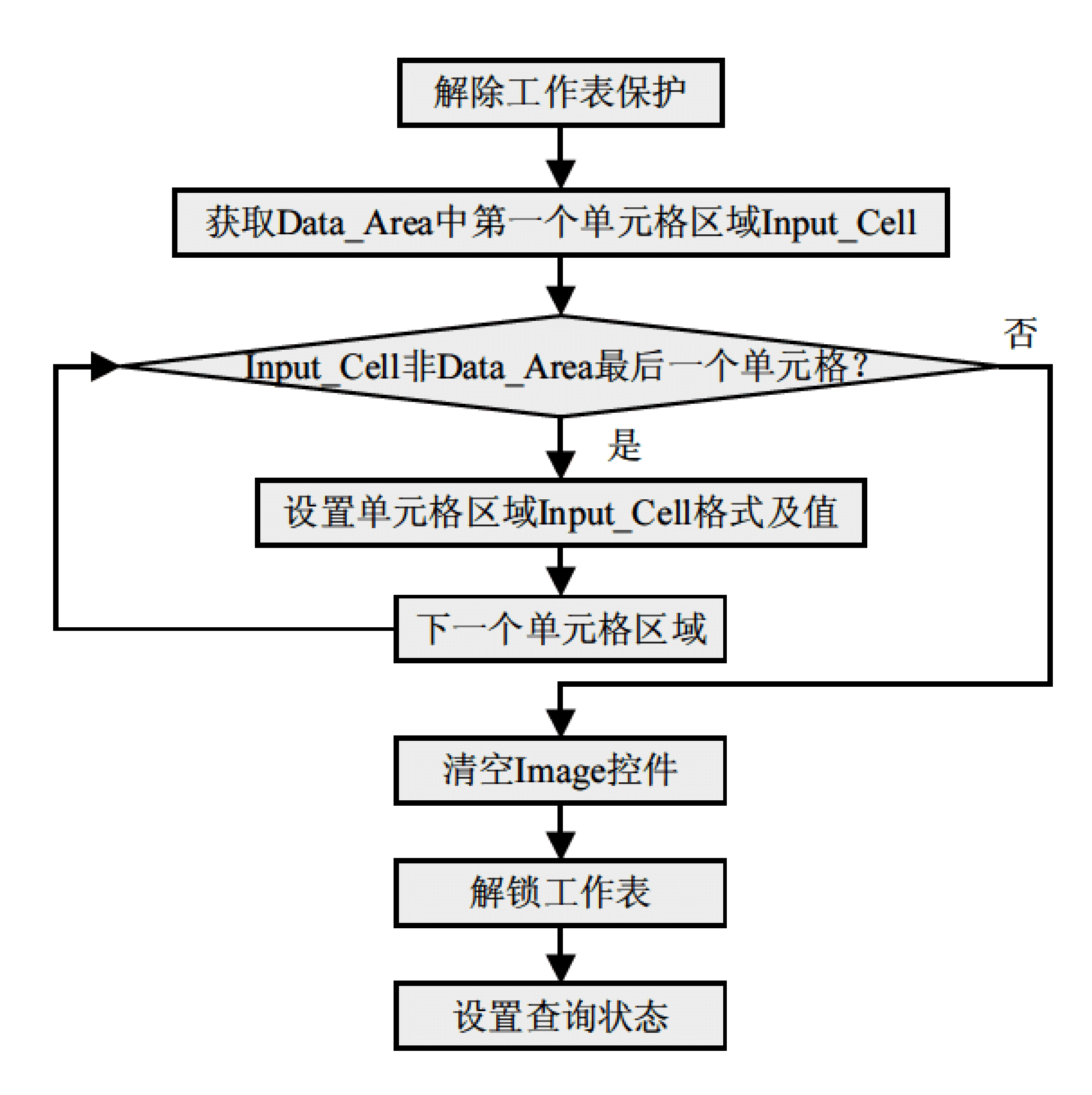


图 8-21　查询记录过程流程图

以下是这两个过程的详细代码解释：

```
'查询初始化过程
Sub DataFind_Status()
    Application.ScreenUpdating = False
    ActiveSheet.Unprotect
    For Each Input_Cell In Range("Data_Area")
        Input_Cell.NumberFormatLocal = "@"
        Input_Cell.Value = ""                           '置空值
    Next
    Sheets("主页").Image1.Picture = LoadPicture("")     '相片为空
    Sheet_Unlock
    Application.ScreenUpdating = True
    Find_Status = True
End Sub

'执行记录查询
Sub Data_Search(myAddress, Find_Value As String)
Dim Find_range As Range
    Application.ScreenUpdating = False              '设置屏幕不更新
    On Error Resume Next
    Set Find_range = Range(myAddress)               '依据传递的参数获取一个单元格对象
    Application.Goto Find_range
    '将库表按照 Find_Range 单元格所在列排序，排序方式按照拼音，顺序为升序
    With Worksheets("库")
         .Range("a1:" & .UsedRange.Address).Sort Key1:=Range(Find_range.Address),
Order1:=xlAscending, Header:=xlGuess, OrderCustom:=1, MatchCase _
         :=False, Orientation:=xlTopToBottom, SortMethod:=xlPinYin, DataOption1:=xlSortNormal
    End With
    With Worksheets("库").Columns(Find_range.Column)
         '在库表的 Find_Range 单元格所在列查找包含 Find_Value 的单元格
```

```
            Set Find_range = .Find(Find_Value, LookIn:=xlValues, SearchOrder:=xlByRows)
            If Not Find_range Is Nothing Then
                '当找到对应单元格后，将该单元格的记录行号保存到参数表中
                Worksheets("参数").Range("a2").Value = Find_range.Row - 1
                '刷新员工档案卡表中的公式
                Sheet_Formula
                '设置非查询状态
                Find_Status = False
                Picture_Load '显示图片
            Else
                '当没找到对应单元格时，继续查询并提示未找到记录
                Find_Status = True
                Range("b2").Select
                MsgBox "对不起,找不到相符的记录,请重新输入"
            End If
        End With
        Application.ScreenUpdating = True
        End
End Sub
```

8.3.10　锁定与解锁工作表过程

员工档案卡工作表在浏览状态下，输入单元格处于锁定状态。当用户浏览员工信息时，需要锁定输入单元格以防用户的误操作，造成数据修改。当用户需要编辑员工信息时，又需要解锁工作表输入单元格。锁定与解锁工作表过程分别完成这两个任务。

这两个过程的流程十分类似，仅仅操作工作表锁定状态时，分别进行相反的设置。这里只对解锁工作表过程加以说明。解锁时，程序首先需要解除工作表的保护，然后依次循环 Input_Area 名称包含的所有单元格。无论单元格是否合并单元格，都设置 Locked 属性为假。为了保证只有那些刚解锁的单元格能编辑，程序还需要重新开启工作表保护，并且设置解锁单元格可选。如图 8-22 所示的是解锁工作表过程的流程图。

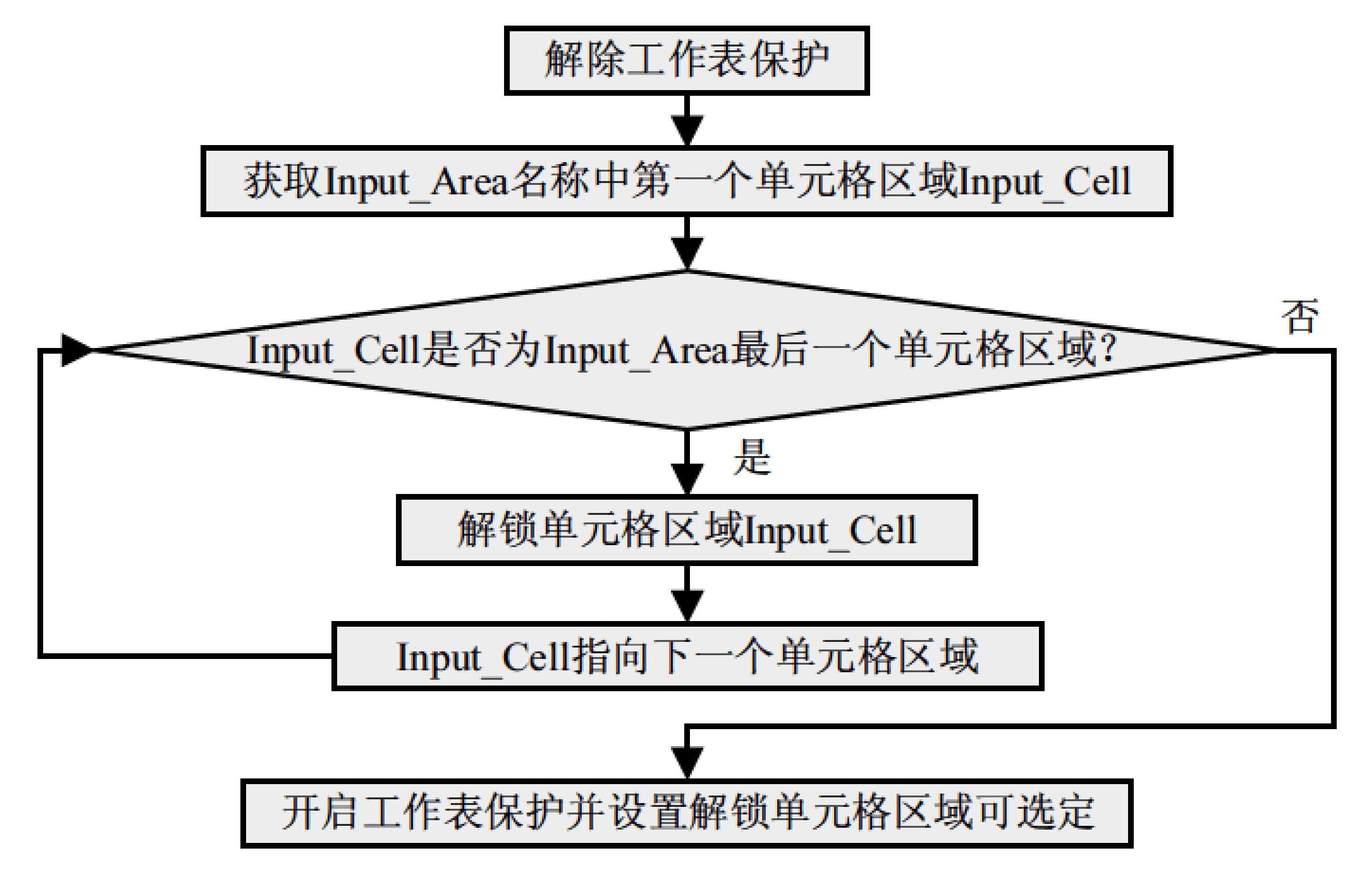


图 8-22　解锁工作表过程流程图

以下是锁定和解锁工作表过程的详细代码解释：

```
'解锁工作表
Sub Sheet_Unlock()
     Range("b2").Select
     '解除工作表保护
     ActiveSheet.Unprotect
     '循环所有需输入的单元格，解除单元格的锁定状态
    For Each Input_Cell In Range("Input_Area")
        If Input_Cell.MergeCells = True Then
            Input_Cell.MergeArea.Locked = False
        Else
            Input_Cell.Locked = False
        End If
    Next
    '重新开启工作表保护
    ActiveSheet.Protect DrawingObjects:=True, Contents:=True, Scenarios:=True
    '设置在工作表中可以被选择单元格只能是已经解除锁定的单元格
    ActiveSheet.EnableSelection = xlUnlockedCells
End Sub

'锁定工作表
Sub Sheet_Lock()
    Range("a1").Select
    ActiveSheet.Unprotect
    For Each Input_Cell In Range("Data_Area")
        If Input_Cell.MergeCells = True Then
            Input_Cell.MergeArea.Locked = True
        Else
            Input_Cell.Locked = True
        End If
    Next
    ActiveSheet.Protect DrawingObjects:=True, Contents:=True, Scenarios:=True
    ActiveSheet.EnableSelection = xlUnlockedCells
End Sub
```

8.3.11　隐藏批注与显示图片过程

隐藏批注与显示图片过程都和员工图片信息相关。在员工档案卡工作表中，用户的图像通过一个 Image 控件显示，当鼠标指针在该控件滑过时，存储员工图片位置的 G2 单元格的批注会显示出来。该批注不会自动隐藏，当鼠标指针移开 Image 控件时，需要将该批注隐藏起来，这就是隐藏批注的功能。

显示图片过程从员工档案卡工作表的 G2 单元格获取员工图片地址，并且按该地址找到该图片加载到 Image 控件中。当 G2 单元格没有任何地址信息时，程序将清除 Image 控件的显示。这两个过程的代码与流程都不复杂，以下不再给出过程的流程图。下面是两个过程的详细代码解释：

```
'隐藏批注
Sub Hidden_Comment()
    '设置任何时候都不显示批注与标识符
    Application.DisplayCommentIndicator = xlNoIndicator
    '设置不显示图像控件所在单元格的批注
    Sheets("员工档案卡").Range("g2").Comment.Visible = False
End Sub

'显示图片
Sub Picture_Load()
    On Error GoTo LoadPicture_Err
    '当没有任何错误发生时，在图像控件中装载 G2 单元格指定路径的图像，否则清空图像控件
    Sheets("员工档案卡").Image1.Picture = LoadPicture(Range("g2").Value)
    Exit Sub
LoadPicture_Err:
    Sheets("员工档案卡").Image1.Picture = LoadPicture("")
End Sub
```

8.4 考勤签到模块代码设计

考勤签到模块用于向当月考勤表登记员工出勤情况。考勤表的样式在前面章节已经介绍，该节将讲述该部分的功能实现。

8.4.1 考勤签到窗体设计

员工签到时可以通过该窗体了解当前的系统时间，从【员工名】下拉列表中选择自己的名字，如果员工名称有重复时，可以参考员工号选择。该窗体的界面如图 8-23 所示。窗体主要包括系统时间显示标签、员工号显示标签、员工名选择列表、签到按钮和关闭窗体按钮。



图 8-23 考勤签到界面

该窗体的制作步骤如下：

（1）在 Excel 2007 的 VBE 开发环境中依次选择【插入】|【用户窗体】命令插入一个用户窗体。随后在【属性】窗口中设置该窗体的名称属性为“frm 考勤”，Caption 属性设置为“考勤签到”。

（2）在控件工具箱中选择标签按钮控件。在窗体中单击鼠标左键并拖动以产生适当大小的标签。随后将该控件再复制 4 份。然后在【属性】窗口中将第一个标签的 Caption 属性设置为“当前时间：”。第二个标签的 Caption 属性设置为空，并将其名称修改为“lab 时间”。

（3）在属性窗口中将第三个标签的 Caption 属性设置为“员工号：”。第四个标签的 Caption 属性设置为空，并将该其名称修改为“lab 员工名”。

（4）在属性窗口中将第五个标签的 Caption 属性设置为“员工名：”。随后在控件工具箱中选择组合框控件。在窗体中单击鼠标左键并拖动以产生适当大小的组合框。然后在【属性】窗口中将其名称设置为“comb 员工名”。Style 属性设置为 2-frmStyleDropDownList，此

项设置将导致该组合框不接受用户输入内容。SelectionMargin 属性设置为 False，此项设置将使该组合框文本区没有留白。

（5）按照步骤（4）相同的方法向窗体再添加两个按钮控件。随后在属性窗口中设置第一个按钮的 Caption 属性为“签到”，名称属性设置为“btn 签到”。第二个按钮的 Caption 属性设置为“关闭”，名称属性设置为“btn 关闭”。

（6）调整各个控件的排列位置并调整窗体大小。将各个控件的位置及大小调整成如图 8-23 所示。

表 8-1 列出了该窗体中与代码相关的各个控件及其功能解释。

表 8-1　考勤签到窗体控件列表

名　称	控件类型	功能解释
lab 时间	标签控件	用于显示当前系统时间
lab 员工号	标签控件	用于显示当前选择的员工的员工号
comb 员工名	复合框控件	用于获取员工名，该列表允许员工名相同，相同的员工名通过员工号加以区分
btn 签到	按钮控件	选择该按钮后，完成当前选择员工的签到工作
btn 关闭	按钮控件	选择该按钮后，退出考勤签到窗体

8.4.2　考勤签到模块执行流程与初始化代码

考勤签到时，员工首先选择员工名，当有重复名时，员工通过员工号校正员工名的选择，直到选择正确的员工名为止。接着单击【签到】按钮，系统根据签到时间，判断员工出勤情况，并将其记录到考勤表中。此处记录的出勤情况包括出勤、旷工和迟到 3 种情况。图 8-24 列出了考勤签到时相应代码的执行过程。

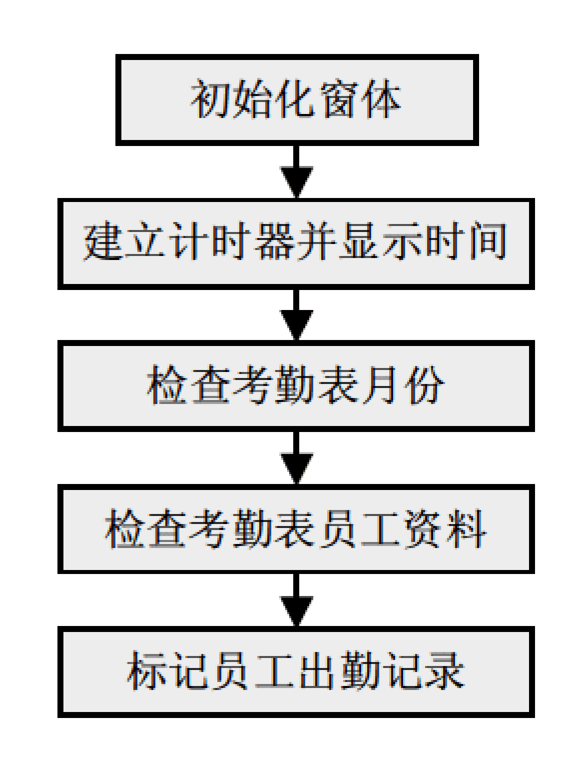


图 8-24　考勤签到窗体程序执行流程图

该流程可以通过窗体的初始化事件过程代码中看出，以下是该窗体的初始化代码：

```
Private Sub UserForm_Initialize()
Dim intRow As Integer
hMain = FindWindow(vbNullString, Me.Caption)          '获取窗口句柄
StartTimer                                            '建立计时器并显示时间到时间标签控件
检查考勤月份                                           '检查考勤月份
考勤表检查                                             '检查考勤表员工资料
intRow = Sheets("库").UsedRange.Rows.Count            '获取库表中已用数据区域行数
comb 员工名.RowSource = "库!B2:B" & intRow            '初始化员工名列表框内容
'下面代码将考勤表激活，并且将活动单元格定位到 A1
Sheets("考勤表").Select
Sheets("考勤表").Range("A1").Select
End Sub
```

代码说明：

- ❑ hMain 公共变量用于获取窗口句柄，该参数将被 StartTimer 所调用，以便于为该窗体

创建一个计时器。

- 系统的考勤表是按照月来记录的，所以当当前日期与考勤表的月份不一致时，应该建立新的考勤表。这就是检查考勤月份过程的工作。
- 考勤表中的员工资料可能与库表中员工资料不一致。这可能是由于员工辞职，库表删除了该员工；也可能是刚有新员工加入，库表添加了新员工资料；也可能是某位员工的员工名输入错误，在库表中修改了该员工的员工名。考勤表检查过程将检查这些，并将考勤表的员工资料加以修正。
- 员工名列表的内容是通过该控件的 Rowsource 属性链接到库表而获得的。指定该属性时，需要指出链接区域的详细位置，此处链接的是库表 B 列中所有员工名区域。链接字符串的格式一般为：表名+!+区域名。
- 当该窗体初始化后，需要激活考勤表，以便于员工了解自己的出勤记录情况。由于系统在生成该表时，可能将活动单元格指定到其他位置，造成显示时位置不正确，所以在激活该表后，修正了活动单元格的位置。

8.4.3 设计计时器代码

窗体中的时间标签动态地显示当前的系统时间，签到者可以根据该时间确定自己当日的出勤情况。标签控件的内容是不会自动更新的，Excel 的窗体也没有定时器功能。为了实现标签控件现实时间每过 1 秒后更新，需要使用定时器，该定时器是通过 API 函数实现的。通常通过 API 函数使用计时器时，一般的方式可参考图 8-25 所示。

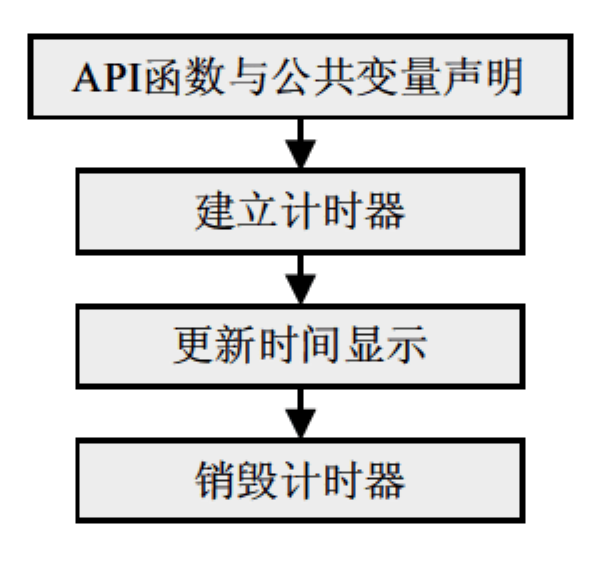


图 8-25　计时器使用过程

下面的代码体现如图 8-25 所示的流程：

```
'API 函数与公共变量声明区域
'SetTimer 函数用于创建一个计时器
Private Declare Function SetTimer Lib "user32.dll" (ByVal hwnd As Long, ByVal nIDEvent As Long, _
    ByVal uElapse As Long, ByVal lpTimerFunc As Long) As Long
'KillTimer 函数用于销毁由 SetTimer 函数创建的计时器
Private Declare Function KillTimer Lib "user32.dll" (ByVal hwnd As Long, ByVal nIDEvetn As Long) As
Long
'FindWindwow 函数用于寻找所有开启窗口中满足指定条件的顶级窗口
Public Declare Function FindWindow Lib "user32.dll" Alias "FindWindowA" (ByVal lpClassName As String, _
    ByVal lpWindowName As String) As Long
'公共变量，主窗口句柄
Public hMain As Long

'更新时间显示过程
'SetTime 过程修改考勤窗体的当前时间标签
Sub SetTime()
frm 考勤.Lab 时间.Caption = Format(Now, "hh:mm:ss")
End Sub
'建立计时器过程
```

```
'StartTimer 过程调用 SetTimer 函数创建一个计时器
Sub StartTimer()
SetTimer hMain, 1001, 1000, AddressOf SetTime
End Sub

'销毁计时器
'EndTimer 过程调用 KillTimer 函数终结计时器
Sub EndTimer()
KillTimer hMain, 1001
End Sub
```

代码说明：

（1）在 API 函数 SetTimer 中出现的 1001 常量参数是生成的计时器的标识，在销毁该计时器时需要使用该参数。

（2）在 API 函数 SetTimer 中出现的 1000 常量参数指定了超时值，1000 即为 1 秒，该参数单位为毫秒。

（3）AddressOf SetTime 语句获取了函数 SetTime 的地址信息。该地址信息被传递给 API 函数 SetTimer 后，将会造成每隔 1 秒时，系统调用一次 SetTime 函数。

（4）上面的代码并不能立即发挥计时器的作用，计时器要正常工作还需要事件的触发。这里所指的事件触发即为窗体的初始化事件，在窗体初始化事件中建立计时器，在计时器工作中执行 SetTime 过程更新时间显示。当窗体关闭时，销毁计时器。与计时器相关的 3 个工作流程都在窗体的相应事件中执行。

8.4.4　设计检查考勤月份代码

在系统中考勤表是按月计算的。当现在所处月份与考勤表记录的月份不一致时，需要将该考勤表保存起来，并且重新建立一新考勤表。这个任务由检查考勤月份过程完成。图 8-26 列出了该过程的流程图。

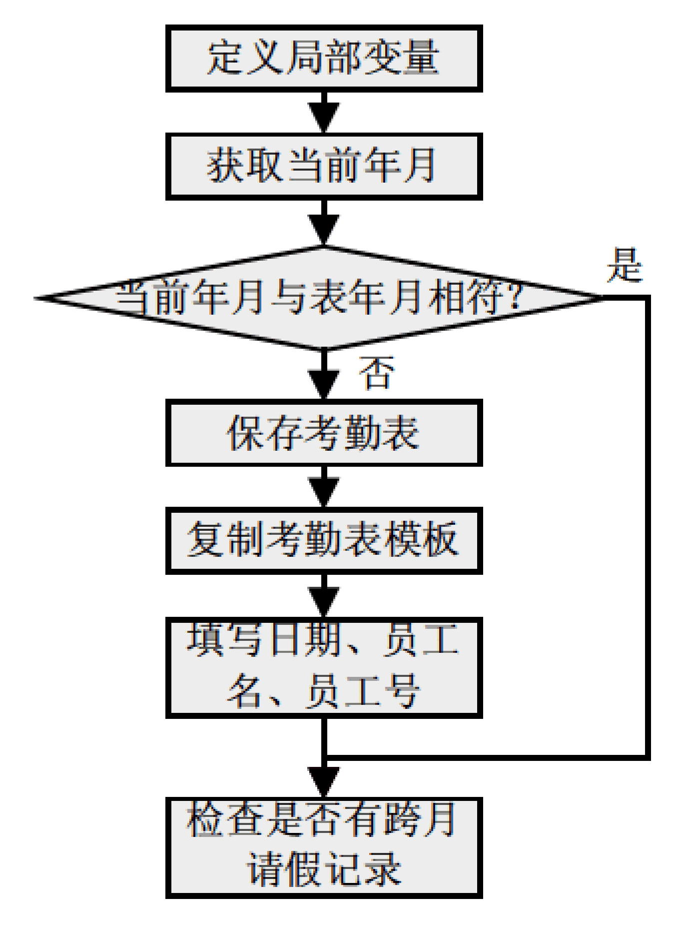


图 8-26　检查考勤月份流程图

该过程的代码如下：

```
'该过程检查考勤表是否与当前月份相符合，当不符合时，建立新的考勤表，旧表重命名保存
Sub 检查考勤月份()
Dim strYear As String, strMonth As String
Dim intRow As Integer, i As Integer, j As Integer
Dim ws1 As Worksheet, ws2 As Worksheet
Dim rg As Range
'获取当前年月
strYear = Format(Now, "yyyy")
strMonth = Format(Now, "mm")
'将考勤表的年月与当前年月比较，不一致时保存该表并重新建立该表
If strYear = CStr(Sheets("考勤表").Range("AH2")) And strMonth = CStr(Sheets("考勤表").Range
("AJ2")) Then
    Exit Sub
Else
    Set ws1 = Sheets("考勤表")
    ws1.Name = "考勤表(" & strYear & "-" & Sheets("考勤表").Range("AJ2") & ")"
    Sheets("考勤表模板").Copy after:=Worksheets("考勤表模板")
    Set ws2 = ActiveSheet
    '定义新考勤表标签名，以及该表记录考勤的年月
    With ws2
        .Name = "考勤表"
        .Range("AH2") = strYear
        .Range("AJ2") = strMonth
    End With
    '获取库表已使用区域行数
    intRow = Sheets("库").UsedRange.Rows.Count
    '将库表中所有员工的员工号、员工名填写到新考勤表
    For i = 2 To intRow
        ws2.Cells(i + 3, 1).EntireRow.Insert                '插入新行
        ws2.Cells(i + 3, 1) = Sheets("库").Cells(i, 1)       '填入员工号
        ws2.Cells(i + 3, 2) = Sheets("库").Cells(i, 2)       '填入员工名
        '复制考勤统计部分的公式
        For j = 34 To 39
            ws2.Cells(i + 4, j).Copy
            ws2.Cells(i + 3, j).PasteSpecial
        Next
        Application.CutCopyMode = False                     '关闭剪切与粘贴模式
    Next
End If
获取旧考勤表最后员工记录行所在的行号
intRow = ws1.UsedRange.Rows.Count -1
'检查上月是否有员工跨月请假，存在时，在当前月份的考勤表中登记并清除旧考勤表的跨月请假记录
For i = 5 To intRow
    If ws1.Range("AO" & i) > 0 Then                         '检查假期统计中的假期剩余
        '在新考勤表中找到该员工的员工号单元格
        Set rg = ws2.Range("A5:A" & intRow).Find(ws1.Range("A" & i))
```

```
            '在新考勤表中标记该员工的请假记录
            For j = 3 To ws1.Range("AO" & i) + 2
                Sheets("考勤表").Cells(intRow, j) = ws1.Range("AN" & i)
            Next
            '清除旧考勤表的假期统计内容
            ws1.Range("AN" & i & ":AO" & i).ClearContents
        End If
Next
End Sub
```

代码说明：

- 首先通过 Format 函数分别以“YYYY”，“MM”格式从当前日期中提取年与月。
- 检查考勤表的年月是否与当前日期的年月相符。如果一致就退出过程，不一致时继续过程。
- 将旧考勤表的标签修改为“考勤表”+“（年-月）”格式，然后从考勤表模板复制建立新的考勤表。命名新表为“考勤表”并填写当前年月到表中的年月单元格。
- 从库表的第二行记录开始，循环到 Sheets("库").UsedRange.Rows.Count 行为止。将所有行中的员工号与员工名写入新考勤表中。在该循环过程中还要重新填写考勤统计区域的公式，这些公式已经在考勤表的最后一行建立。例如，第五行的出勤所在列的公式为“=COUNTIF($C5:$AG5,"√")”，关于该公式的意义，参见前面知识点五。
- 在旧考勤表中可能记录了部分跨月请假时，部分没有标记请假记录的信息，这部分信息需要在当前月份下标记。这时，需要根据旧考勤表中记录的请假类型和假期剩余信息在新考勤表中加以标记。
- 标记跨月请假时，从旧考勤表的第五行开始，循环到最后一员工所在行。当 ws1.Range("AO" & i) > 0 为真时，可以判断有跨月请假记录，然后在新考勤表中找到对应员工。通过一个内循环，从新考勤表的第三列开始，一直到假期的终结日，将所有该员工在这些日期内的考勤记为 ws1.Range("AN" & i)单元格内容。
- 最后清除该员工的假期统计信息。

8.4.5　设计检查考勤表员工资料代码

考勤表和库表中相应的员工号与员工名可能存在部分出入。这些可能是由于员工辞职，库表删除了该员工；也可能是刚有新员工加入，库表添加了新员工资料；也可能是某位员工的员工名输入错误，在库表中修改了该员工的员工名。检查考勤表过程将检查这些问题，并将考勤表的员工资料加以修正。

在处理这些问题时，有一种情况不需要追究，即库表删除了某员工。不需要在考勤表中删除该员工，因为可能在该月的前一时间段里该员工的考勤记录仍然是真实的。该过程的流程图如图 8-27 所示。

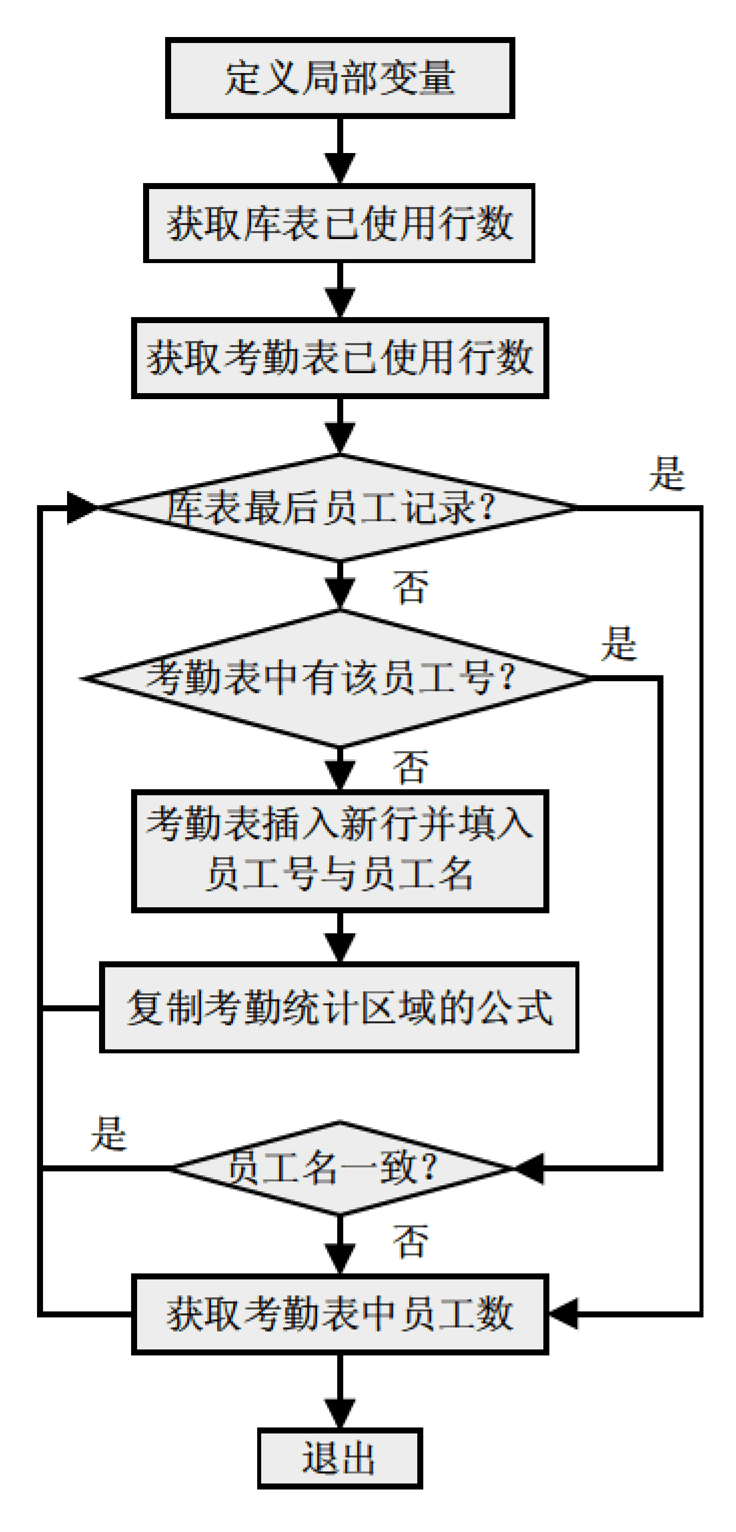


图 8-27　检查考勤表流程图

该过程的代码如下：

```
'该过程检查考勤表和库表的员工信息是否一致，当新增员工时，在考勤表中建立该员工资料
Sub 考勤表检查()
Dim intRow1 As Integer, intRow2 As Integer, i As Integer
Dim intEnd As Integer, intStart As Integer, j As Integer
Dim rgFind As Range
'获取库表中已使用行数
intRow1 = Sheets("库").UsedRange.Rows.Count
'获取考勤表中已使用行数
intRow2 = Sheets("考勤表").UsedRange.Rows.Count
'循环检测对比，将库表中新添加的员工填入考勤表中
'对于有删除员工的情况，这里不需要在考勤中删除对应资料，因为可能该员工在当前月份可能工作过一段时间
For i = 2 To intRow1
    '在考勤表中查找对应员工的员工号
    Set rgFind = Sheets("考勤表").Range("A5:A" & intRow2).Find(Sheets("库").Range("A" & i))
    '当没有找到该员工号时，将该员工资料建立到考勤表中
    If rgFind Is Nothing Then
        Sheets("考勤表").Cells(i + 3, 1).EntireRow.Insert
        Sheets("考勤表").Cells(i + 3, 1) = Sheets("库").Cells(i, 1)
        Sheets("考勤表").Cells(i + 3, 2) = Sheets("库").Cells(i, 2)
        intRow2 = intRow2 + 1
        For j = 34 To 39
            Sheets("考勤表").Cells(i + 4, j).Copy
            Sheets("考勤表").Cells(i + 3, j).PasteSpecial
        Next
```

```
            Application.CutCopyMode = False
        Else
            '当已有该员工号时，但员工名不一致时，修改员工名即可
            If rgFind.Offset(0, 1) <> Sheets("库").Cells(i, 2) Then
                rgFind.Offset(0, 1) = Sheets("库").Cells(i, 2)
            End If
        End If
Next
End Sub
```

代码说明：

- ❑ 首先该过程从库表和考勤表中获取了两表中已使用行数，这两个变量在后续循环比较过程中被调用。
- ❑ 从库表的第二行开始，循环到该表的 intRow1 行。当在考勤表中找不到该员工时，插入新行，并在考勤表中建立该员工的资料，包括员工号与员工名。为该员工的考勤统计区域复制统计公式，然后继续下一个员工行。
- ❑ 当考勤表中查到该员工号但是员工名不一致时，修改该员工的员工名后继续下一个员工行。

8.4.6　设计标记员工出勤代码

标记员工出勤的工作，包括标记正常出勤、迟到和旷工。前两项是由签到按钮完成，签到按钮不涉及旷工情况。本小节将分别讲述【签到】按钮和记旷工的功能实现。

本系统的默认最后签到时间为早上 9 点。在该时间前签到的员工为正常出勤，否则为迟到。对于没有签到的员工，退出窗口时，系统将提示是否记为旷工。单击窗体上的【签到】按钮时，将触发签到按钮单击事件过程，该过程的流程图如图 8-28 所示。

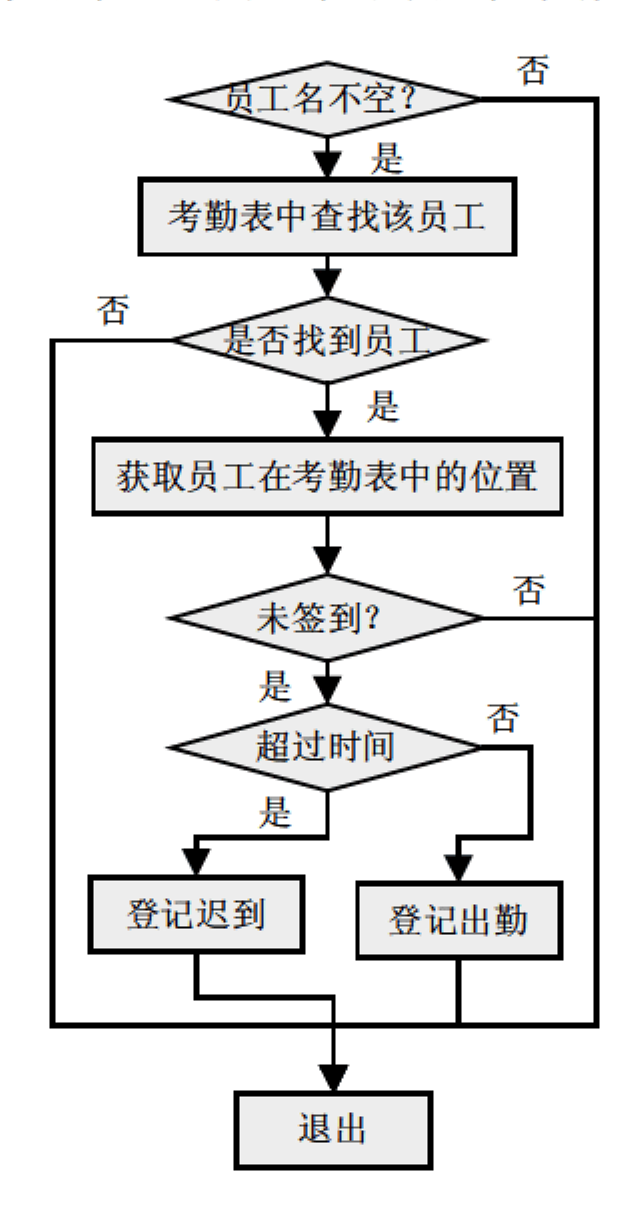


图 8-28　【签到】按钮单击事件过程流程图

该过程代码如下：

```
Private Sub btn 签到_Click()
Dim strName As String, rg As Range
Dim intRow As Integer, j As Integer
strName = lab 员工号.Caption                                '获取员工号
'当没有选择员工时，退出过程
If Len(strName) = 0 Then Exit Sub
'在考勤表中查询员工号
Set rg = Sheets("考勤表").Range("A5").EntireColumn.Find(strName)
'没有找到该员工资料时退出过程
If rg Is Nothing Then
    MsgBox "考勤表中没有员工" & strName & "的资料！", vbOKOnly + vbInformation, "提示"
    Exit Sub
Else
    intRow = rg.Row
    j = Day(Now) + 2
    If Len(Sheets("考勤表").Cells(intRow, j)) Then
        MsgBox "员工" & strName & "已经完成签到！", vbOKOnly + vbInformation, "提示"
    Else
        If Time > TimeValue("09:00:00") Then
            Sheets("考勤表").Cells(intRow, j) = "★"          '标记为迟到
        Else
            Sheets("考勤表").Cells(intRow, j) = "√"          '标记为出勤
        End If
    End If
End If
End Sub
```

代码说明：

- 此处在查询员工时使用的是员工的员工号，而不是姓名。因为姓名有可能一致，而员工号不会重复。
- 程序从窗口的员工号标签控件获取员工号，然后根据员工号字符串长度确定窗体是否获得了员工名输入。当没有输入员工名时，退出过程，然后在考勤表中查询该员工号。找到该员工时，首先检测该员工是否已经完成签到。当有签到时再检测签到时间是否超过时间，针对不同情况，标记员工出勤为迟到或出勤。

对于在规定签到时间没有签到的员工，需要标记为旷工。这项工作是在退出签到窗体时，进行提示操作的。退出窗口时，程序首先获取考勤表使用区域行数并计算当前日期对应的表格列号，然后程序依次循环考勤表考勤数据区域各行，查找当前日期下没设置考勤信息的员工。当发现有未设置考勤的员工时，提示是否记该员工旷工。当用户确定时，程序调用记旷工过程完成标记员工旷工操作。

记旷工过程和窗口退出过程的操作大体一致，只是在找到有员工没有记考勤时，直接将该员工的考勤记为“■”即旷工。因而下面在给出流程图时，只给出窗口的 QueryClose 事件过程流程图。如图 8-29 所示的是 QueryClose 事件的流程图。

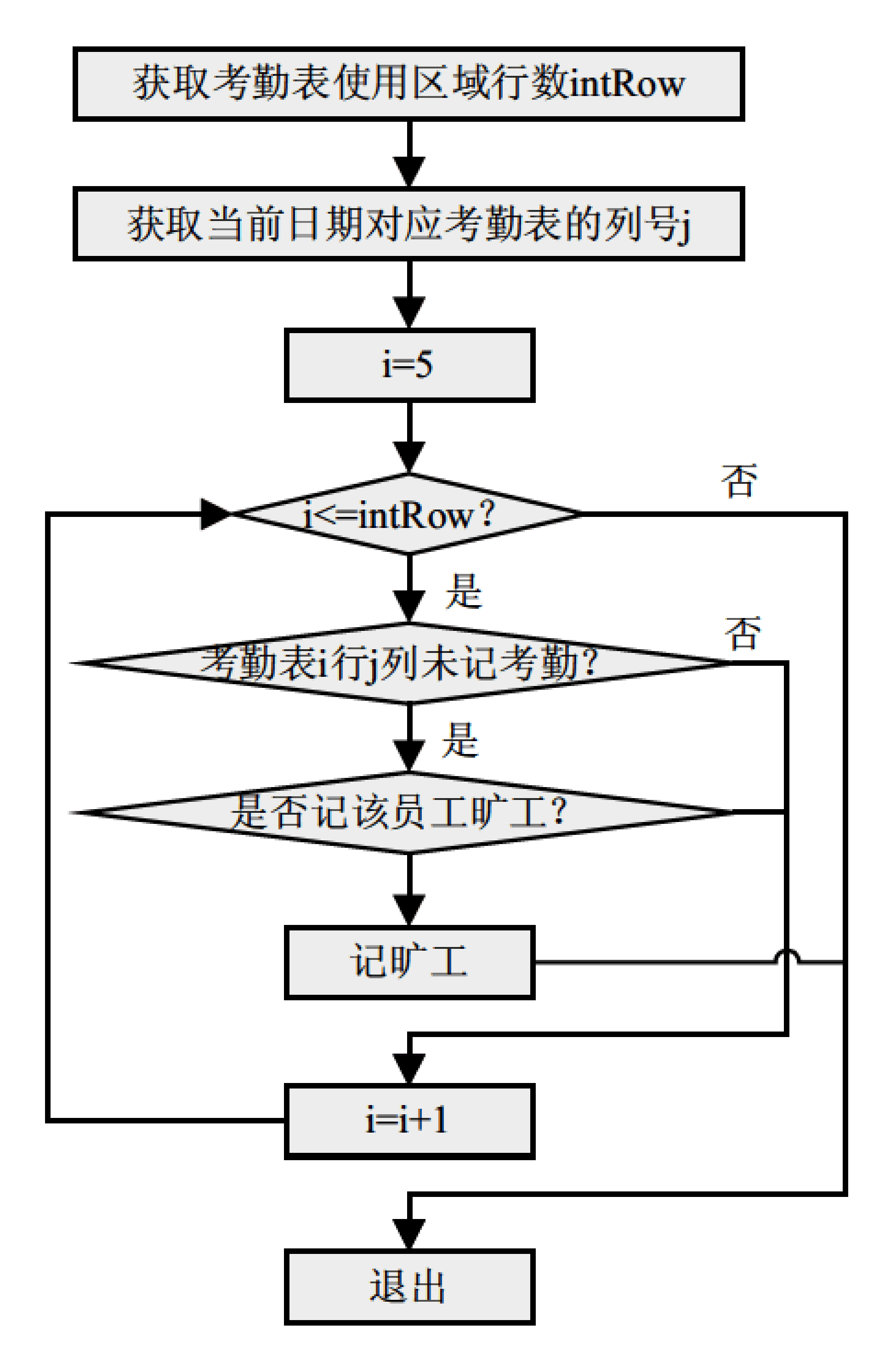


图 8-29　QueryClose 事件过程流程图

从上面的流程图中可以看出程序在调用记旷工过程后，直接终止了过程，该跳转避免无意义的循环。在窗口的 QueryClose 事件代码中通过循环来检测是否存在员工没有记考勤。这里只需要检测到一个员工未记考勤即可，不需要每个都确认一次，而在记旷工过程中则遍历了所有员工的考勤记录。所以在执行完记旷工过程后没有必要再继续检查下去，而是直接终止了过程。

记旷工过程的代码如下：

```
Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
Dim i As Integer, intRow As Integer
'取得考勤表使用区域行数
intRow = Sheets("考勤表").UsedRange.Rows.Count -1
'计算当前日期对应的表格列号
j = Day(Now) + 2
For i = 5 To intRow
    If Sheets("考勤表").Cells(i, j) = "" Then
        i = MsgBox("所有未签到员工是否记未旷工？", vbOKCancel + vbQuestion, "提示")
        If i = vbYes Then
            记旷工
        End If
        EndTimer
        Exit Sub
    End If
Next
EndTimer
```

```
End Sub

Sub 记旷工()
Dim intRow As Integer, i As Integer, j As Integer
intRow = Sheets("考勤表").UsedRange.Rows.Count -1
j = Day(Now) + 2
For i = 5 To intRow
    With Sheets("考勤表")
        If .Cells(i, j) = "" Then
            .Cells(i, j) = "■"
        End If
    End With
Next
End Sub
```

QueryClose 事件过程代码说明：

- ❑ 考勤签到窗口关闭时，需要检查是否有员工没有签到。当有未签到员工时，提示是否将这些没有签到员工的当天考勤记为旷工。
- ❑ 局部变量 j 是当天的日期在考勤表中的列号，当天所有员工的考勤记录都记录在该列下。
- ❑ 局部变量 i 是当前考察的员工记录所在的行号。头条员工记录位于考勤表的第 5 行，所以初始值为 5。因为考勤表中最底部的一行是保存的行格式，没有实际记录值，所以 i 的结束值为考勤表已使用区域的行数减 1。

记旷工过程代码说明：

过程首先获得考勤表的最后一行有数据区域的行数，然后根据日期确定当日考勤记录的列位置，最后循环判断各个员工是否已经签到，对于没有任何签到记录的员工标记为旷工记号。

8.4.7 设计窗体其他功能代码

该模块所有的重要功能都已经实现，但是还有一部分代码也要实现的。这部分代码语句简单，没有包含在大的功能中，包括窗体关闭事件、员工名的改变事件以及窗体显示过程。

窗体关闭事件过程完成两个任务：一是将活动表重新定位到主页表上，另一个是卸载窗体。该事件的代码如下：

```
Private Sub btn 关闭_Click()
Sheets("主页").Select
Unload Me
End Sub
```

员工名改变事件用来同步员工号标签控件的显示。它根据用户选择的员工名所在的索引位置获取对应的员工号。该过程代码如下：

```
Private Sub comb 员工名_Change()
Lab 员工号.Caption = Sheets("库").Range("A" & comb 员工名.ListIndex + 2)
End Sub
```

考勤签到过程连接主页表上的【考勤签到】按钮。该过程用于打开考勤签到窗体。该过程的代码如下：

```
Sub 考勤签到()
frm 考勤.Show
End Sub
```

该过程要连接到主页表上的【考勤签到】按钮，其步骤如下：

（1）激活主页表，右击【考勤签到】按钮。

（2）在弹出的快捷菜单中选择【指定宏】命令。

（3）打开【指定宏】对话框，在【宏名】的下拉列表框中选择【考勤签到】宏，单击【确定】按钮即可，如图 8-30 所示。



图 8-30　为【考勤签到】按钮指定宏

8.5　请假登记模块代码设计

在考勤签到模块中只能记录员工正常出勤、迟到和旷工 3 种考勤情况。当员工需要请假时，考勤签到模块将无法完成。请假登记模块就是用来完成该功能的，它由一个请假登记窗体来实现。该窗体的界面如图 8-31 所示。

该窗体将完成两部分的任务：一是将员工从起始日到结束日的考勤记为相应的请假类型，二是将这些信息登记到请假登记表中。



图 8-31　请假登记窗体界面

8.5.1　请假登记窗体设计

窗体上存在的控件在前面设计考勤登记窗体时都有讲述，这里对于窗体中如何添加控件的步骤不再加以说明。窗体上的控件数量比较多，表 8-2 详细列出了各个将在代码中使用到的控件。

表 8-2　请假登记窗体控件列表

控 件 名 称	控 件 类 型	说　　明
frm 请假登记	窗体	即请假登记窗体，在该窗体完成请假登记工作
lab 员工号	标签控件	该控件类似于考勤登记中的员工号控件，用于校正员工名的选择
comb 员工名	复合框控件	该控件链接自库表，显示了所有员工的姓名。该列表允许出现重复名称
comb 请假类型	复合框控件	该控件用于获取员工请假的类型
comb 起始年	复合框控件	该控件用于获取请假起始日的起始年份。该列表只包含两个年份，一个是当前的年份，一个是当前年的后年年份。因为请假不可能超过一年
comb 起始月	复合框控件	该控件用于获取请假起始日的起始月份。该列表的内容从 1 到 12
comb 起始日	复合框控件	该控件用于获取请假起始日的起始日期。列表将根据当前月份确定到底该月有多少天，然后再生成列表内容
comb 结束年	复合框控件	用于获取请假结束日的结束年份。同 comb 起始年控件类似
comb 结束月	复合框控件	用于获取请假结束日的结束月份。同 comb 起始月控件类似
comb 结束日	复合框控件	用于获取请假结束日的结束日期。同 comb 起始日控件类似
btn 确定	按钮控件	选择该按钮将根据窗体输入数据完成员工请假登记工作
btn 取消	按钮控件	退出窗体

上述请假类型复合框控件包含了病假、事假与出差 3 种情况。起始日和结束日都是通过 3 个符合框控件来获取的。起始日和结束日用当前日完成初始化。

8.5.2　窗体初始化

整个窗体的大部分工作都在【确定】按钮单击事件中完成。有少部分事件，例如员工名复合框改变事件、月复合框改变事件等，它们都是为了完成相应辅助输入员工请假信息而设置的。而在整个窗体正常运作之前，窗体完成了对自己的初始化工作。初始化完成的工作主要是为各个复合框控件获取对应的列表数据。

以下代码是该窗体的初始化代码：

```
Private Sub UserForm_Initialize()
Dim intRow As Integer
'获取请假类型列表框数据
With comb 请假类型
        .Clear
        .AddItem "事假"
        .AddItem "病假"
        .AddItem "出差"
End With
'获取员工名列表框数据
intRow = Sheets("库").UsedRange.Rows.Count
comb 员工名.RowSource = "库!B2:B" & intRow
'获取年月日各列表框数据
初始化年月日
End Sub
```

代码说明：

- 程序首先为请假类型列表复合框获取列表数据。在该步骤中，首先清除该复合框数据，以免造成潜在的列表数据内容重复问题。
- 然后程序获取了库表的可用数据行数，根据该行数确定了所有员工的员工名在库表中的范围。使用复合框的 RowSource 属性直接获取员工列表数据。
- 年月日列表框的初始化稍微复杂，初始化过程调用了自定义过程完成该工作。相关年月日复合框的设计内容参见 8.5.3 节。

8.5.3　年月日复合框相关代码设计

在窗口中包含了很多与年月日相关的设置控件，窗口初始化时，这些控件的内容页需要随之被初始化。另外，用户在选择了对应的年份和月份后，该月份下的日复合框内的日期数据需要随新年份和月份发生变化。要完成这些工作都需要本节介绍的过程执行相应的任务。

下面的初始化年月日过程用于设置窗口中所有与年月日相关的复合框控件的项目。在该过程中需要设置 4 个复合框的项目，这些复合框分别是起始年、结束年、起始月和结束月。两个年复合框只需要当前年份和次年年份即可，月复合框的项目是固定的 1~12。如图 8-32 所示的是该过程的详细流程图。

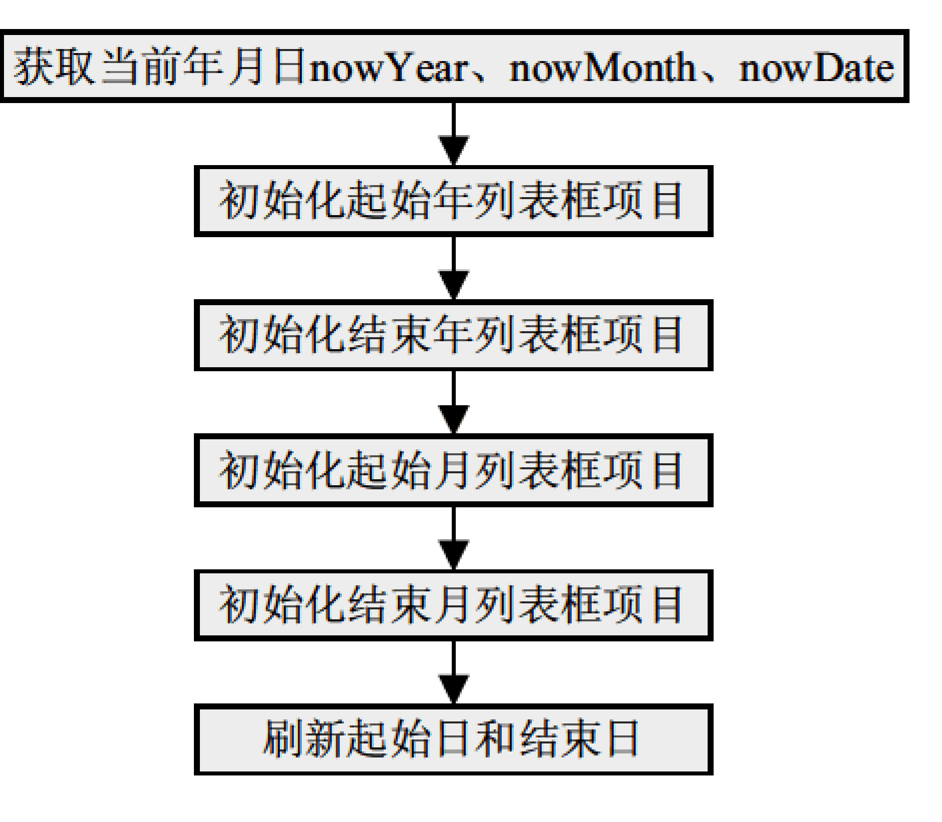


图 8-32　初始化年月日复合框流程图

下面的程序代码是年月日初始化自定义过程：

```
Sub 初始化年月日()
Dim nowYear As String, nowMonth As String, nowDate As String
Dim i As Integer
'获取当前年月日数据
nowYear = Format(Now, "YYYY")
nowMonth = Format(Now, "M")
nowDate = Format(Now, "D")
'初始化起始年列表框数据
With comb 起始年
        .Clear
        .AddItem nowYear
        .AddItem CStr(CInt(nowYear) + 1)
        .Value = nowYear
End With
'初始化结束年列表框数据
With comb 结束年
        .Clear
        .AddItem nowYear
        .AddItem CStr(CInt(nowYear) + 1)
        .Value = nowYear
End With
```

```
'初始化起始月列表框数据
With comb 起始月
    .Clear
    For i = 1 To 12
        .AddItem CStr(i)
    Next
    .Value = nowMonth
End With
'初始化结束月列表框数据
With comb 结束月
    .Clear
    For i = 1 To 12
        .AddItem CStr(i)
    Next
    .Value = nowMonth
End With
'初始化起始日与结束日列表框数据
刷新起始日
刷新结束日
comb 起始日.Value = nowDate
comb 结束日.Value = nowDate
End Sub
```

代码说明：

- 在过程的前面部分，程序获取了当前日期的年月日数据，这部分数据将被初始化过程作为初始数据。即当完成初始化工作后，在年月日中显示的年月日即为当前日期。
- 初始化年列表框时，首先清除了列表框的内容，然后向该列表框中写入了两个年份，分别是当前年份以及次年的年份。因为记请假时年份的跨度不可能超过两个年份。
- 初始化月列表框比较简单。通过一个循环将 1~12 的整数写入即可。因为每一年都包含了 12 个月。
- 初始化日列表框使用了自定义过程。因为刷新日列表框过程中的代码可以在年份和月份的变更事件中重复使用。刷新日列表框的相关代码见下面的介绍。

当窗体初始化或者年份与月份列表框数据发生变更时将会触发刷新起始日和结束日列表框过程。在刷新起始日和刷新结束日过程中，首先需要计算起始日或结束日所在月份的总天数，然后通过一个 For 循环将该月份各个日期添加到列表框中。在计算某月总天数时是通过计算当月 1 号与次月的 1 号之间的差确定的。当起始日或结束日为 12 月某日时，此时的次月是次年的 1 月份，而其他情况下次月都是起始月或结束月加 1 即可。两个过程的流程大体类似，如图 8-33 所示的是刷新起始日过程的流程。

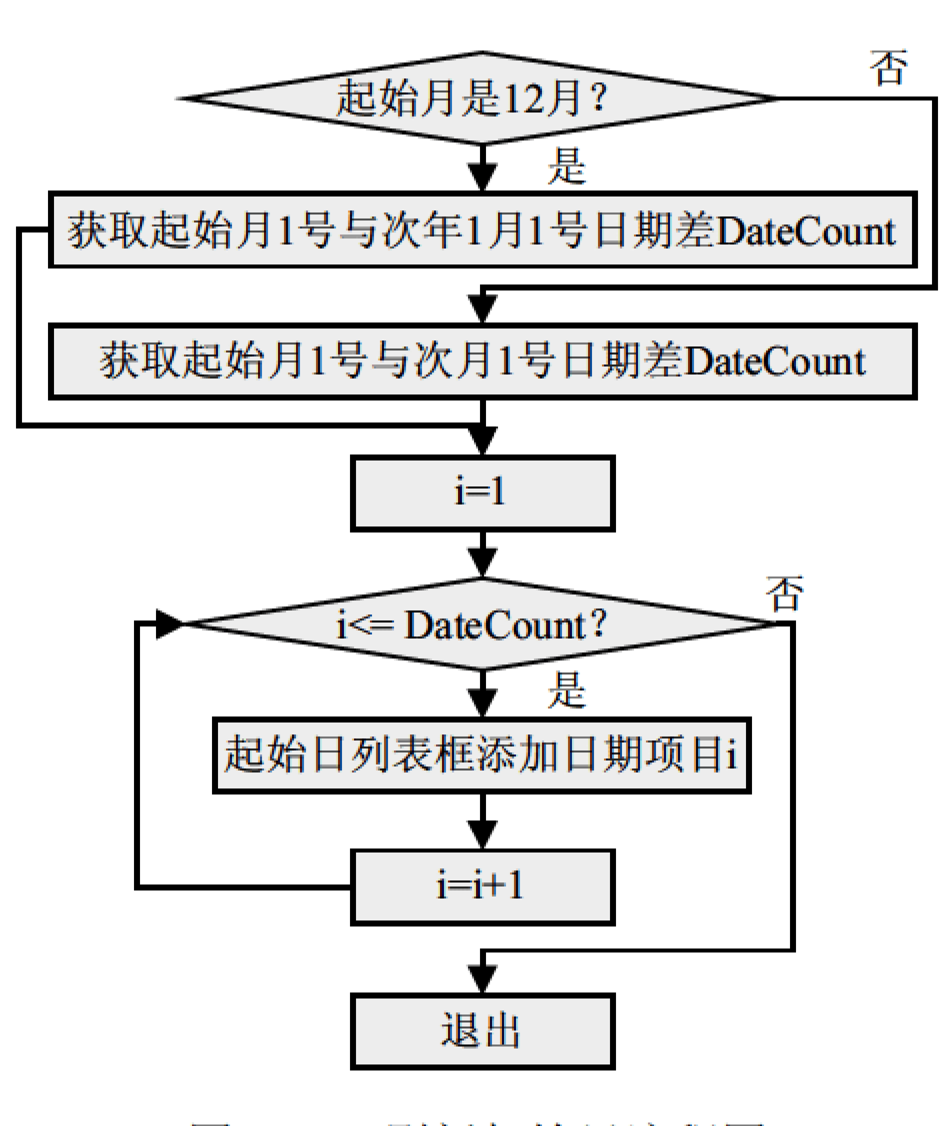


图 8-33　刷新起始日流程图

以下是这两个过程的详细代码解释：

```
Sub 刷新起始日()
Dim DateCount As Integer, i As Integer
'根据不同情况计算当前起始年与起始月下的日期数
If CInt(comb 起始月) = 12 Then
    DateCount = DateDiff("d", comb 起始年 & "-" & comb 起始月 & "-1", CStr(CInt(comb 起始年) + 1)
& "-1-1")
Else
    DateCount = DateDiff("d", comb 起始年 & "-" & comb 起始月 & "-1", comb 起始年 & "-" &
CStr(CInt(comb 起始月) + 1) & "-1")
End If
'刷新日列表框数据
comb 起始日.Clear
For i = 1 To DateCount
    comb 起始日.AddItem i
Next
End Sub

Sub 刷新结束日()
Dim DateCount As Integer, i As Integer
'根据不同情况计算当前结束年与结束月下的日期数
If CInt(comb 结束月) = 12 Then
    DateCount = DateDiff("d", comb 结束年 & "-" & comb 结束月 & "-1", CStr(CInt(comb 结束年) + 1)
& "-1-1")
Else
    DateCount = DateDiff("d", comb 结束年 & "-" & comb 结束月 & "-1", comb 结束年 & "-" &
CStr(CInt(comb 结束月) + 1) & "-1")
End If
'刷新结束日列表框数据
comb 结束日.Clear
For i = 1 To DateCount
    comb 结束日.AddItem i
Next
```

代码说明：

- 起始日与结束日列表框数据确定的方法都是一样的，因此两个自定义过程的代码基本上是一致的。
- 要确定日列表框的数据，唯一需要确定的就是当前选择的年月下该月的日数。这里确定的方法是通过当前月份的 1 号日期与次月的 1 号对比获得（关于 DateDiff 函数的介绍见 8.1.6 节），但是当当前选择月份为 12 月份时，计算次月的方式发生很大变化，因为次月是次年的 1 月份，所以这里使用了 If 语句。
- 在获得对应的日期数后，日列表框的数据就很容易被确定下来。这里通过一个循环将从 1 到该日期数的所有整数写入了该日列表框。

前面讲到在月份列表框数据发生变更时将会触发刷新起始日和结束日列表框过程。这些过程的代码都很简单，仅仅触发相应日列表框的刷新自定义过程，这里不再列出详细的程序说明。这些事件的过程代码如下：

```
Private Sub comb 结束年_Change()
刷新结束日
End Sub

Private Sub comb 结束月_Change()
刷新结束日
End Sub

Private Sub comb 起始年_Change()
刷新起始日
End Sub

Private Sub comb 起始月_Change()
刷新起始日
End Sub
```

8.5.4 确认请假登记代码设计

在确认请假登记过程中需要完成的工作比较多，包括检查列表框控件内容是否空、检查日期差是否超额、检查起始日是否位于结束日后等众多步骤。这些步骤中大部分都是并不复杂的判断，只有判定员工跨月请假时需要多进行些处理。如图 8-34 所示的是该过程的流程图。

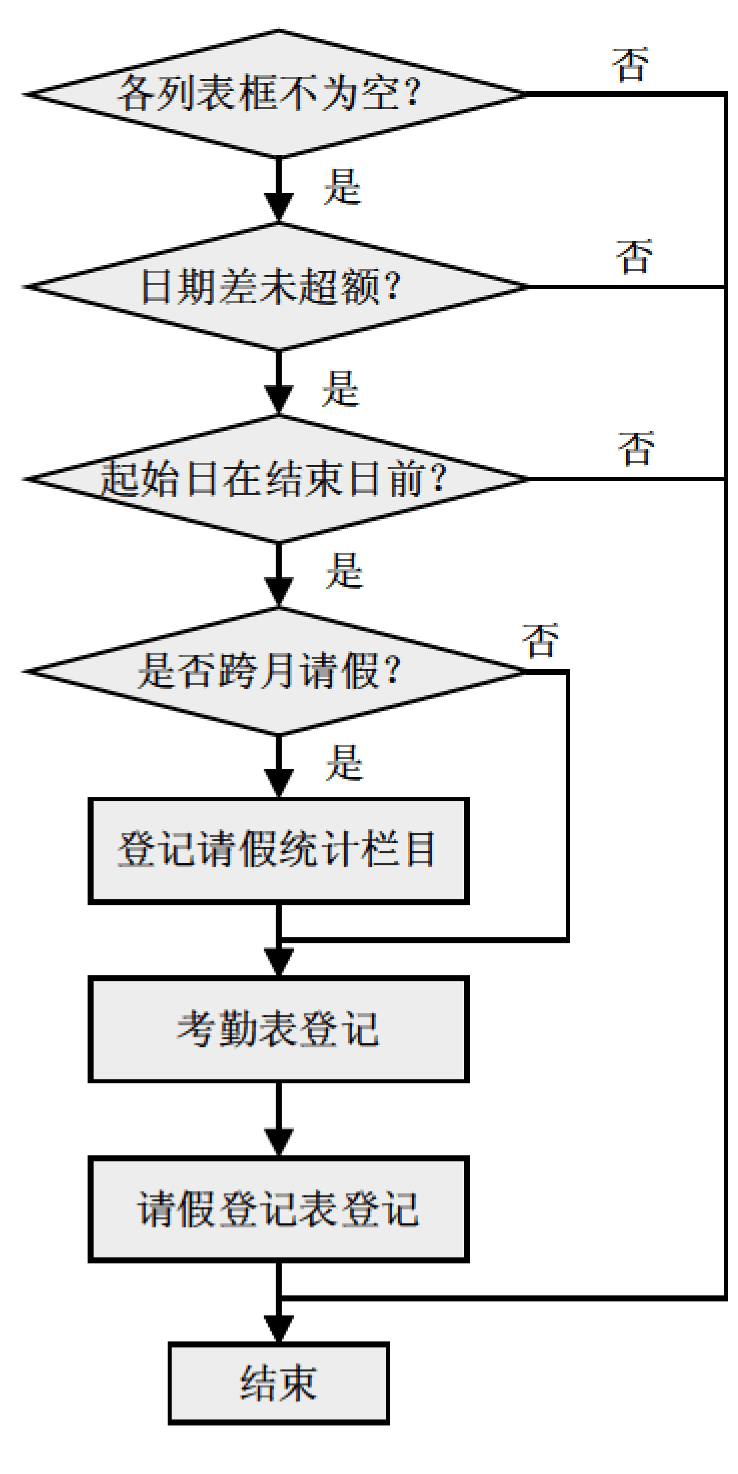


图 8-34　确认请假登记流程图

```
Private Sub btn 确定_Click()
Dim i As Integer, j As Integer, DateCount As Integer
```

```
Dim intRow As Integer, rg As Range
'检查各个列表框控件数据是否为空
If comb 员工名.Value = "" Then
    MsgBox "没有选择员工名！", vbOKOnly + vbInformation, "提示"
    Exit Sub
End If
If comb 请假类型.Value = "" Then
    MsgBox "没有选择请假类型！", vbOKOnly + vbInformation, "提示"
    Exit Sub
End If
If comb 起始年.Value = "" Then
    MsgBox "没有选择起始年！", vbOKOnly + vbInformation, "提示"
    Exit Sub
End If
If comb 起始月.Value = "" Then
    MsgBox "没有选择起始月！", vbOKOnly + vbInformation, "提示"
    Exit Sub
End If
If comb 起始日.Value = "" Then
    MsgBox "没有选择起始日！", vbOKOnly + vbInformation, "提示"
    Exit Sub
End If
If comb 结束年.Value = "" Then
    MsgBox "没有选择结束年！", vbOKOnly + vbInformation, "提示"
    Exit Sub
End If
If comb 结束月.Value = "" Then
    MsgBox "没有选择结束月！", vbOKOnly + vbInformation, "提示"
    Exit Sub
End If
If comb 结束日.Value = "" Then
    MsgBox "没有选择结束日！", vbOKOnly + vbInformation, "提示"
    Exit Sub
End If
'计算起始日与结束日间的日期差
DateCount = DateDiff("d", comb 起始年 & "-" & comb 起始月 & "-" & comb 起始日, comb 结束年 & "-"
& comb 结束月 & "-" & comb 结束日)
'当日期差是否超过 31 天
If DateCount > 31 Then
    MsgBox "起始日与结束日间不能超过 31 天！", vbOKOnly + vbInformation, "提示"
    Exit Sub
End If
'检查结束日是否在在起始日前
If DateCount < 0 Then
    MsgBox "结束日不能在起始日前！", vbOKOnly + vbInformation, "提示"
    Exit Sub
End If
'根据员工号在考勤表中找到该员工
Set rg = Sheets("考勤表").Range("A5").EntireColumn.Find(Lab 员工号.Caption)
If rg Is Nothing Then
    MsgBox "该员工在考勤表中不存在！", vbOKOnly + vbInformation, "提示"
```

```
        Exit Sub
Else
        intRow = rg.Row
End If
'检查是否存在跨月请假情况
If CInt(comb 起始月) <> CInt(comb 结束月) Then
        DateCount = DateDiff("d", comb 起始年 & "-" & comb 起始月 & "-1", comb 起始年 & "-" &
CStr(CInt(comb 起始月) + 1) & "-1")
        For i = CInt(comb 起始日) To DateCount
                Select Case comb 请假类型
                        Case "事假"
                                Sheets("考勤表").Cells(intRow, i + 2) = "▲"
                                Sheets("考勤表").Cells(intRow, 40) = "▲"
                        Case "病假"
                                Sheets("考勤表").Cells(intRow, i + 2) = "△"
                                Sheets("考勤表").Cells(intRow, 40) = "△"
                        Case "出差"
                                Sheets("考勤表").Cells(intRow, i + 2) = "C"
                                Sheets("考勤表").Cells(intRow, 40) = "C"
                End Select
                j = j + 1
        Next
        DateCount = DateDiff("d", comb起始年 & "-" & comb起始月 & "-" & comb起始日, comb结束年 &
"-" & comb 结束月 & "-" & comb 结束日)
        Sheets("考勤表").Cells(intRow, 41) = DateCount -j + 1
Else
        For i = CInt(comb 起始日) To CInt(comb 结束日)
                Select Case comb 请假类型
                        Case "事假"
                                Sheets("考勤表").Cells(intRow, i + 2) = "▲"
                        Case "病假"
                                Sheets("考勤表").Cells(intRow, i + 2) = "△"
                        Case "出差"
                                Sheets("考勤表").Cells(intRow, i + 2) = "C"
                End Select
        Next
End If
'在请假登记表中记录该员工的请假记录
With Sheets("请假登记表")
        intRow = .UsedRange.Rows.Count
        .Cells(intRow, 1).EntireRow.Insert
        .Cells(intRow, 1) = Lab 员工号.Caption
        .Cells(intRow, 2) = comb 员工名.Value
        .Cells(intRow, 3) = comb 请假类型.Value
        .Cells(intRow, 4) = comb 起始年 & "-" & comb 起始月 & "-" & comb 起始日
        .Cells(intRow, 5) = comb 结束年 & "-" & comb 结束月 & "-" & comb 结束日
End With
End Sub
```

代码说明：

❑ 程序首先检测各个列表框控件内容是否为空，以确保后面的代码以及相关的操作能

获取到数据。

- 在该程序中，默认的请假天数不能超过 31 天，在程序中需要检测日期差是否超过这个额度。
- 对于起始日和结束日有先后顺序，这个先后顺序是由 DateCount 的正负情况确定的。
- 考勤表是逐月记录的。当出现跨月请假情况时，需要将请假的类型以及剩余天数记录在请假统计的请假类型和假期剩余里。这些数据将在下一个月产生新的考勤表时被调用并清除。当请假的起始与结束年月一致时，只需要将请假类型登记到当前月即可。
- 当完成了考勤表的登记后，还需要将该员工的请假记录登记到请假登记表中。该表记录了员工请假的流水账。

8.6　系 统 测 试

本系统包含 3 个部分，本节测试部分将对系统的 3 个功能部分分别加以测试，具体包括员工资料登记、员工考勤登记和员工请假登记。这里 3 个测试都针对同一个员工。

8.6.1　员工资料登记

（1）在主页单击【员工资料管理】按钮，系统自动激活员工档案卡工作表。此时该工作表中显示的是最后一位员工的信息。此处需要建立新员工的信息，用户只需要在加载项菜单中单击【增加】按钮即可。在员工档案卡表中建立新员工信息如图 8-35 所示。

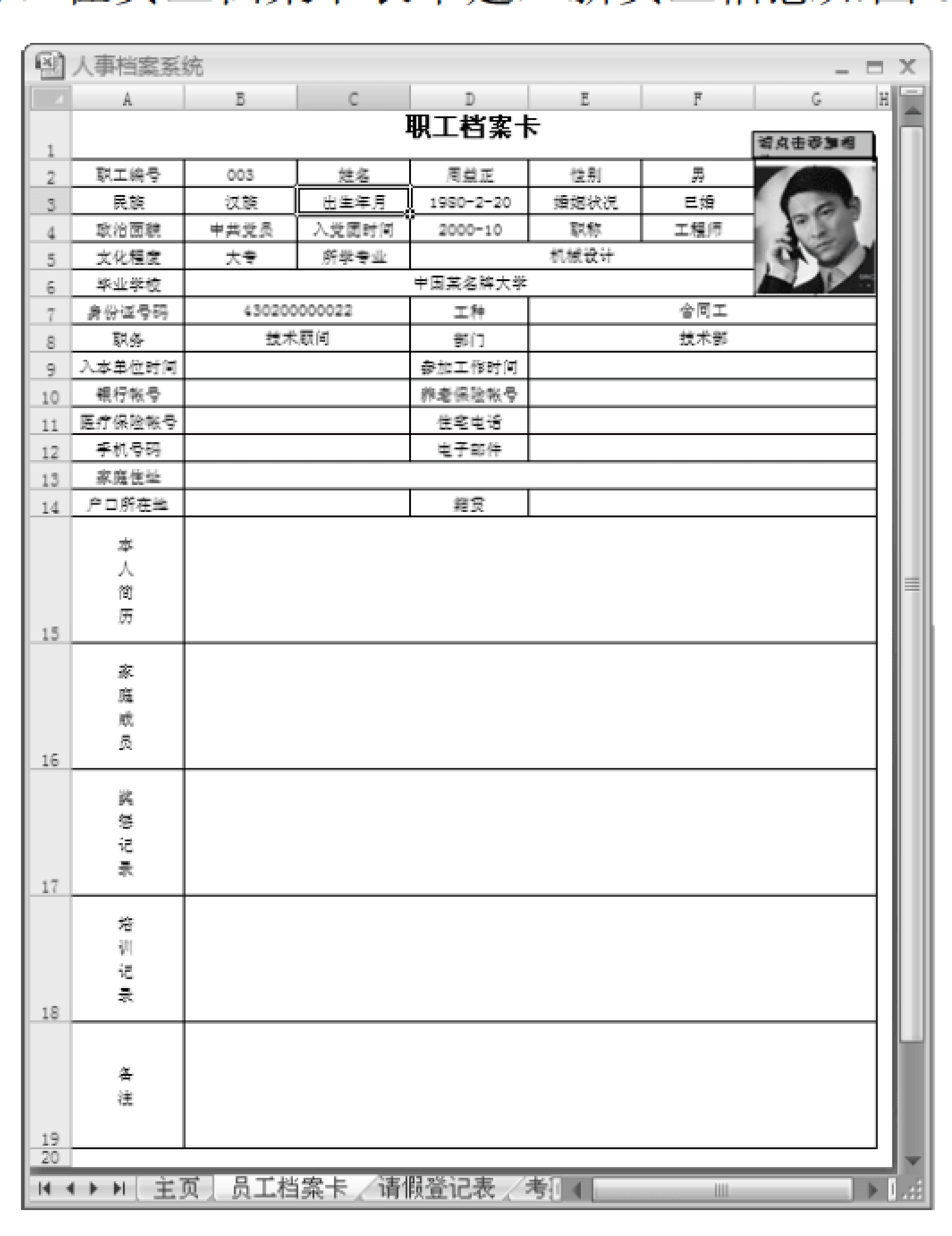


图 8-35　新员工资料建立

（2）建立好该员工的信息资料后，在加载项菜单中单击【保存】按钮。此时新员工的资料信息将被保存到库中。打开库表可以查看到该条刚建立的员工信息，如图 8-36 所示。



图 8-36　库表员工信息

8.6.2　员工考勤登记

在主页单击【考勤签到】按钮，在随后打开的【考勤签到】对话框中选择员工名周益正，如图 8-37 所示。单击【签到】按钮，由于已经超过了时间，系统认定该员工为迟到，如图 8-38 所示。

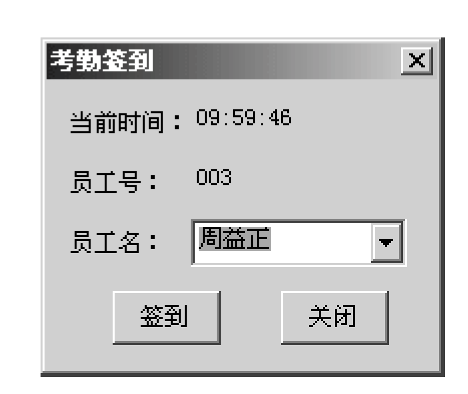


图 8-37　【考勤签到】对话框



图 8-38　考勤签到结果

8.6.3　员工请假登记

在主页单击【请假登记】按钮，在随后弹出的【请假登记】对话框中设置员工名、请假类型和时间跨度，如图 8-39 所示。最终在考勤表中该员工当日的出勤信息被以实心三角板标记，如图 8-40 所示。员工的所有请假信息页将被登记到请假登记表中，该表的结果如图 8-41 所示。



图 8-39　请假登记设置



图 8-40　登记员工请假信息

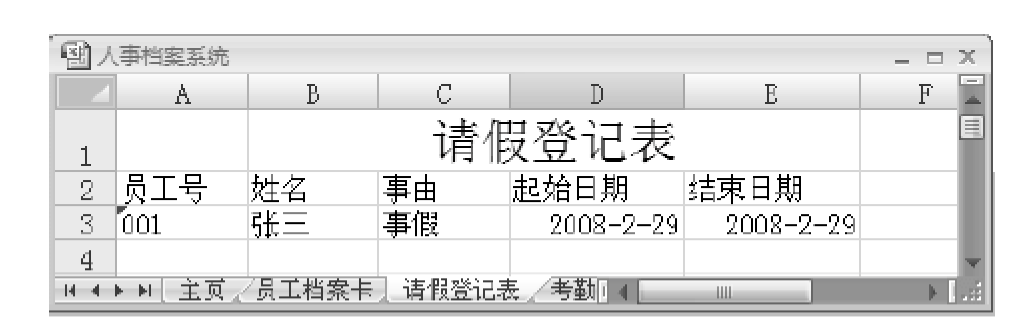


图 8-41　请假登记表

第 9 章　商场销售数据管理系统

对于大部分企业，商品销售管理可以使用比较大众化的软件，但很多时候企业的商品都具有自己的特殊性，用户希望能够定制适合自己的管理软件。本系统即为一个实例，它被开发用于针对连锁商场电器商品销售数据记录与统计的管理。该系统的个性很强，对于希望开发自己的管理系统的用户是一个借鉴。

9.1　系 统 概 论

系统所针对的对象仅仅是连锁商场的电器商品。系统功能包括基本数据建立、商品销售登记、商品销售查询与统计。本系统是一个数据与程序完全分离的系统，所有的数据资料都通过单独的文件保存，并且基本数据和商品销售数据也是单独存放。

设计思路

本系统以 Excel 为操作平台，使用 Access 文件保存数据，通过 DAO 对象模型实现 Excel 程序文件与 Access 数据文件间的数据互动。值得一说的是，在本书中共有 3 个例子讲述了这种前台程序文件与后台数据文件的互动。在前面的客户管理系统中，采用了 Excel 数据保存与 Access 文件保存相结合的方式，在这个章节读者可以初步了解 DAO 对象的使用。本章实例则完全使用 DAO 对象模型，这里读者可以完全了解该对象的使用，后面的合同管理系统将使用 DAO 对象模型实现相同功能。

本系统的功能包括基本数据建立、商品销售数据登记、商品销售数据查询与分析 3 大功能。用户在操作时完全通过窗体实现交互。该系统的功能结构如图 9-1 所示。

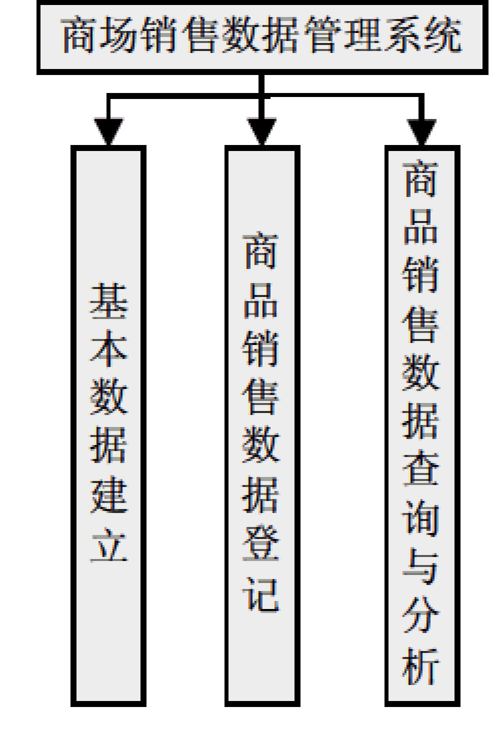


图 9-1　商场销售数据管理系统功能结构图

以下是系统中用到的所有窗体的大致介绍。

- ❑ 基本信息设置窗体：窗体名称为 frmSetup，在该窗体中可以浏览与编辑已经建立的基本资料。这些资料包括商场名、商品品牌与规格信息。
- ❑ 商品销售数据登记窗体：窗体名称为 frmInput，在该窗体中可以输入商品销售数据，包括商场名、品牌名称、商品规格、尺寸、销售数量、销售单价。该窗体是一个多用途窗体，它不仅可以在销售商品时使用，也可以在查询数据时修改数据。

- 查询销售数据设置窗体：窗体名称为 frmQuery，在该窗体中可以设置各种详细的查询选项。在商品销售数据登记窗体中包含的项目都出现在了这些查询选项中。当该窗体中没有设置任何项目的查询条件时，默认的查询方式即为查询所有销售数据。
- 查询条件编辑窗体：窗体名称为 frmFilterEdit，在该窗体中可以重新编辑查询条件。该窗体不仅可以编辑最终查询条件，还可以单独编辑某一项的查询条件。
- 查询显示窗体：窗体名称为 frmQryResult，在该窗体中显示了满足查询条件的所有销售记录。用户可以修改这些数据也可以导出数据。

9.2　数据表设计

在该系统中数据与前台程序代码是分开存放的。另外，不同的数据也是分开保存的。基本资料数据和商场销售数据资料分开保存在两个数据文件中，这些文件都是 Access 文件。为便于后面前台窗体与程序设计部分的讲述，本章首先介绍数据表的建立。

值得注意的是这些数据文件都保存在 DB 文件夹中，该文件夹应该与 Excel 文件位于同一文件夹中。基本信息资料表的名称是 Info.mdb，商品销售数据资料表的名称为 DB.mdb。

9.2.1　基本信息资料表设计

在基本信息资料表文件中有 3 个表，分别是 t_MarketInfo、t_MarkInfo 和 t_ProductInfo。使用 Access 2007 打开该文件后，可以在表列表框中看到如图 9-2 所示的情况。

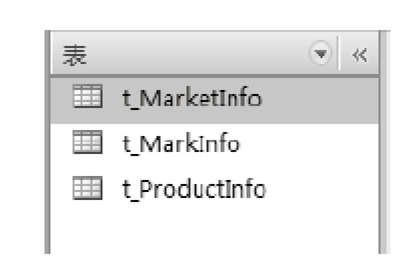


图 9-2　基本信息资料表中的表

下面是各个表的详细介绍。

- 表 t_MarketInfo：存储商场名信息。仅有一个字段即商场名称，字段的数据类型为文本。图 9-3 与图 9-4 分别显示了该表的设计视图与数据表视图格式。

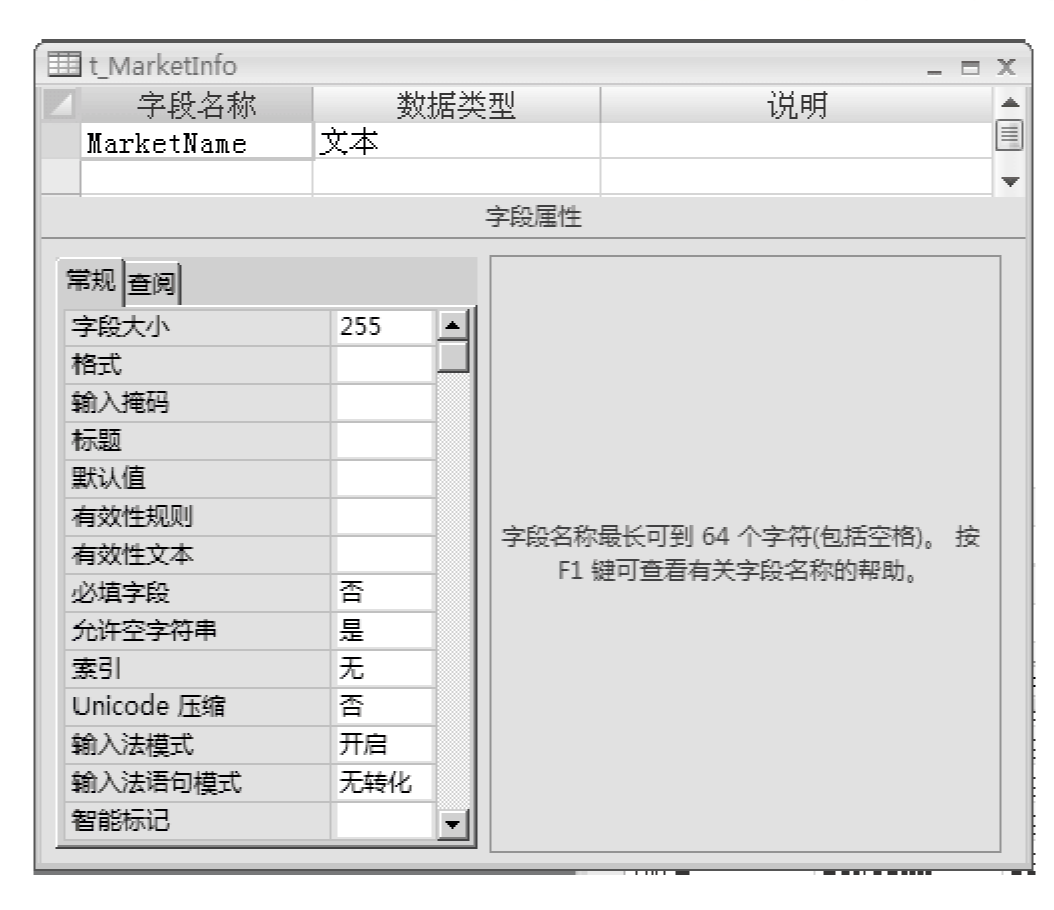


图 9-3　表 t_MarketInfo 设计视图



图 9-4　表 t_MarketInfo 数据表视图

- 表 t_MarkInfo：该表中存储的是品牌名称。仅包含一个字段即品牌名称，字段的数据

类型为文本。图 9-5 与图 9-6 分别是该表的设计视图与数据表视图格式。该表与表 t_ProductInfo 建立了对应关系，因而其数据表视图与表 t_MarketInfo 的数据表视图存在差别。要查看该关系，可以首先进入设计模式然后在设计模式标题上右击，在弹出的快捷菜单中选择【关系】命令即可。图 9-7 与图 9-8 分别显示了打开关系窗口的方式与关系窗口的界面。



图 9-5　表 t_MarkInfo 设计视图

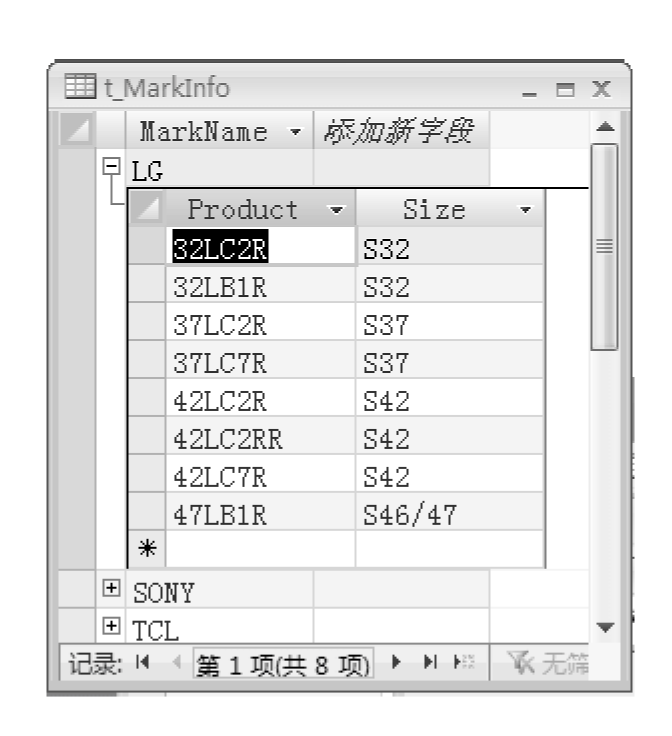


图 9-6　表 t_MarkInfo 数据表视图

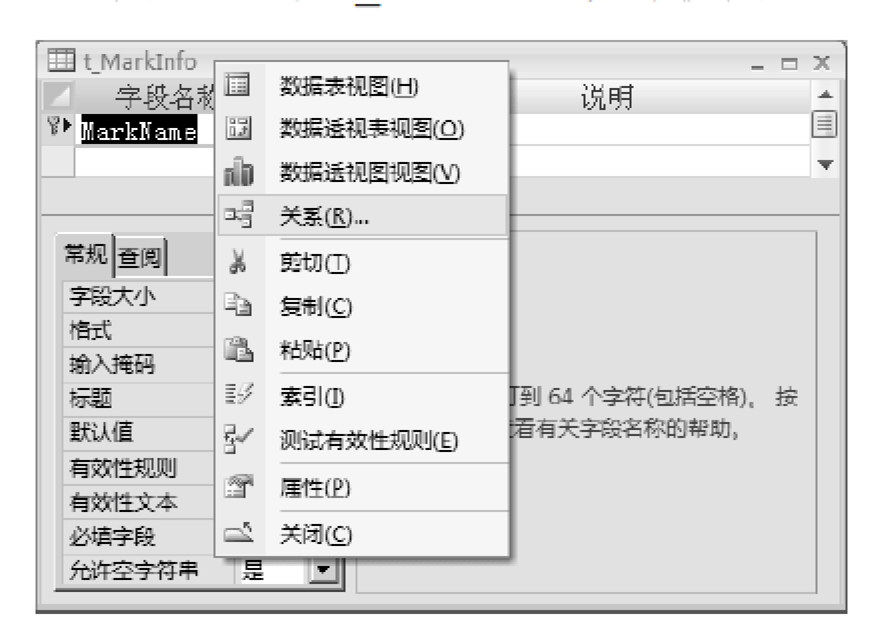


图 9-7　在设计视图中打开关系窗口

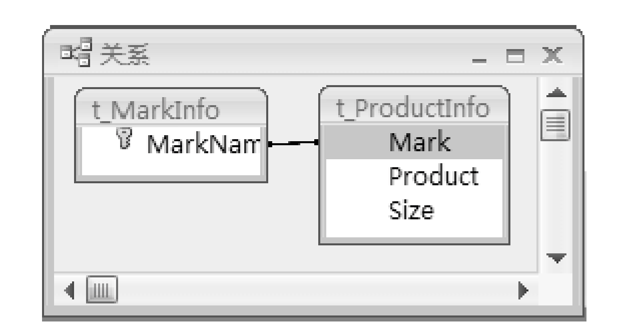


图 9-8　表关系窗口

- t_ProductInfo：该表存储的是商品的相关信息。包含的 3 个字段分别是品牌名称（文本类型）、商品名称（文本类型）和商品规格（文本类型）。图 9-9 与图 9-10 分别显示了该表的设计视图与数据表视图格式。

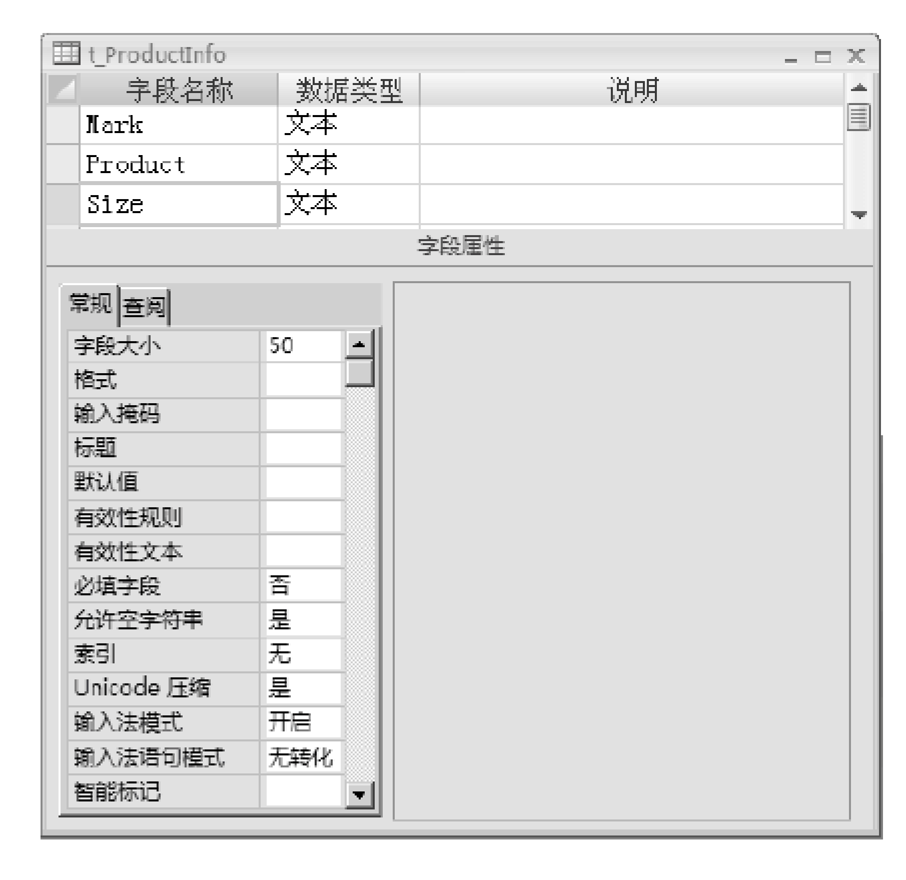


图 9-9　表 t_ProductInfo 设计视图

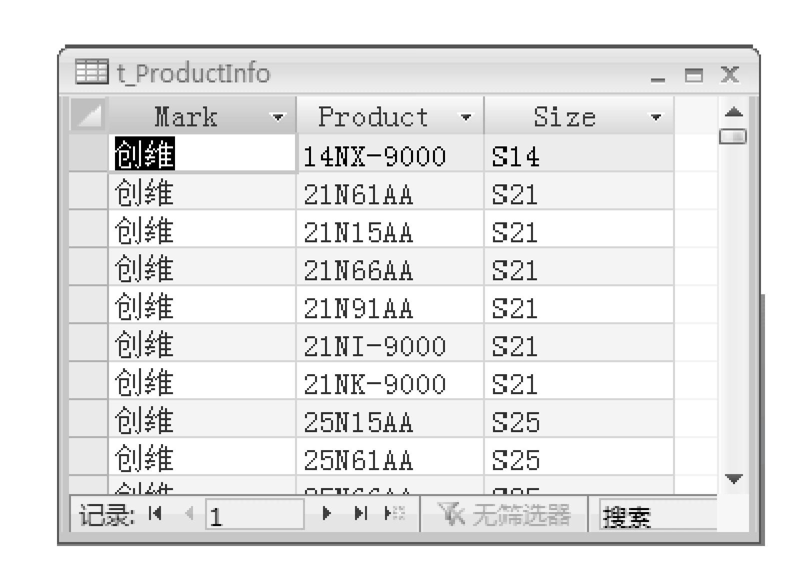


图 9-10　表 t_ProductInfo 数据表视图

9.2.2　商品销售数据资料表设计

商品销售数据资料表用于存储明细的商品销售数据，该资料表仅包含了一个表即 t_Sale。图 9-11 与图 9-12 分别显示了该表的设计视图与数据表视图格式。表 t_Sale 的结构建立方式在系统中不是通过手动建立的。在系统中有段代码负责建立该表的结构，读者可以参考该段代码学习如何通过 DAO 对象在 Access 表中建立新表的方式。该段代码将在后续章节加以介绍。



t_Sale

字段名称	数据类型	说明
MarketName	文本	
Mark	文本	
Product	文本	
Size	文本	
Number	数字	
Price	数字	
Date	数字	
ID	文本	

字段属性

常规　查阅

字段大小	50
格式	
输入掩码	
标题	
默认值	
有效性规则	
有效性文本	
必填字段	否
允许空字符串	否
索引	无
Unicode 压缩	否
输入法模式	开启
输入法语句模式	无转化
智能标记	

字段名称最长可到 64 个字符(包括空格)。按 F1 键可查看有关字段名称的帮助。

图 9-11　表 t_Sale 设计视图



t_Sale

MarketNam	Mark	Product	Size	Number	Price	Date	ID
大润发春申	LG	32LB1R	32	2	1000	8	20070508122027
大润发春申	LG	32LB1R	32	3	200	8	20070508122033
大润发春申	LG	32LB1R	32	10	20000	8	20070508122038
大润发闸北	长虹	CHD32300	32	40	10000	8	20070508122104
大润发闸北	康佳	LCTM2011S	20	30	2999	8	20070508122118
家乐福万里	厦华	LC32HU25	32	10	1000	16	20070508122133
大润发春申	创维	21N15AA	21	2	33	29	20070529193611

记录: 第 1 项(共 7 项)　无筛选器　搜索

图 9-12　表 t_Sale 数据表视图

9.3　公共模块代码设计

在系统中存在部分公共变量与公用函数，这些公共变量和公共函数不能被保存在窗体中，但是它们又必须被窗体调用。系统的公共模块正是为了保存这些公共变量与公共函数而设立的。系统建立了两个公共模块，分别用来保存公共变量和公共函数与过程。

9.3.1　公共变量模块设计

在公共变量模块存储了系统中使用到的所有公共变量。这些变量将在系统运行过程中被

初始化并反复调用。以下是定义这些公共变量的代码：

```
'全局变量
'--------------------------------------------------------------------------------
Public db As DAO.Database                    '数据库对象
Public strMarketFilter As String             '存储对商场的单项筛选条件
Public strMarkFilter As String               '存储对品牌的单项筛选条件
Public strProductFilter As String            '存储对产品型号的单项筛选条件
Public strSizeFilter As String               '存储对尺寸的单项筛选条件
Public strNumberFilter As String             '存储对数量的单项筛选条件
Public strPriceFilter As String              '存储对单价的单项筛选条件
Public strDateFilter As String               '存储对日期的单项筛选条件
Public intFilterIndex As Integer             '决定操作是针对哪个单项的
Public strSQL As String                      '总筛选条件
Public opMethod As Boolean                   '运算方式
Public isEditRecord As Boolean               '判断是否在修改记录
Public arrEdited As Variant                  '记录被编辑记录的数据
Public itemEdited As ListItem                '被编辑的记录
Public isNeedUpdate As Boolean               '是否需要更新列表
Public isSaveAll As Boolean                  '表示保存结果的范围（总查询记录集或选择后的记录集）
```

代码说明：

大部分的变量从代码注释中就可以了解该变量的意义，例如那些单项筛选条件，这部分变量记录的正是查询窗口中每一个单项的查询条件，而总筛选条件则是对单项筛选条件的综合。还有部分变量单从注释中无法详细了解它们的功用。这里将对这些变量加以说明：

- intFilterIndex 整型变量：该变量记录的是当前用户设置的查询条件是属于哪个单项的。例如当单击了某个单项查询条件的【修改】按钮时，这个变量将会被设置为该项的对应序号。编辑查询条件窗口根据该变量确定显示和编辑操作是针对哪个单项的。
- opMethod 布尔变量：该变量被用在查询条件设置窗口中，它记录了当前设置的运算方式。对于每个单项，可以设置查询条件的运算方式。例如：当需要查询两个商场的所有销售记录时。在查询设置窗口中，首先选择第一个商场名，并确认运算方式是否运算，单击该项对应的【添加】按钮，第一个查询商场条件即被设置。然后再次选中第二个商场，单击【添加】按钮。此时单击【编辑】按钮时就会出现类似以下内容的条件“MarketName='大润发春申' or MarketName='国美二店'”。
- isNeedUpdate 布尔变量：该变量用于确认是否重新显示查询结果窗体中显示的记录数据。当完成记录的编辑工作后，并退出销售记录编辑窗体时，显示在查询结果窗体上的数据需要立即更新。如果在销售记录编辑窗中没有进行修改或删除操作，此时没有必要更新结果显示。
- isSaveAll 布尔变量：在查询结果窗口中，可以有针对性地选择已查询到的结果记录，并且将该结果导出到新的 Excel 文件中。该变量正是用于标识当前的数据导出是全部查询结果导出还是部分结果导出。

9.3.2　启动窗体公共过程代码设计

公共函数与过程模块中包含的过程与函数比较多，但是可以划分为几个部分，它们是启动窗体过程、总查询字符串修改过程、数据库建立与更新过程、压缩数据库过程。启动窗体过程被自定义菜单调用，通过自定义菜单可以访问对应的窗体。本小节主要讲解启动窗体过程的代码设计，随后的小节将依次讲解剩下的各个过程。以下是该过程的代码：

```
'启动窗体过程
Sub RunInputForm()                          '启动输入信息窗体
frmInput.Show
End Sub

Sub RunQueryForm()                          '启动查询窗体
frmQuery.Show
End Sub

Sub CreateBaseInfo()                        '启动基本数据建立系统
frmSetup.Show
End Sub
```

代码说明：

上述启动窗体的过程包含的语句都十分简单，这些过程名在建立自定义菜单时被使用，选择对应的菜单即会调用对应的过程打开对应的窗体。

9.3.3　总查询字符串设置过程

在查询设置窗口中，当所有的查询单项的条件都设置完后（当然也可以不设置部分单项，程序会自动加以识别），程序将通过该过程产生总查询字符串。只有获得了该字符串后才能获得满足条件的数据库查询记录集。该自定义过程的流程图如图 9-13 所示。

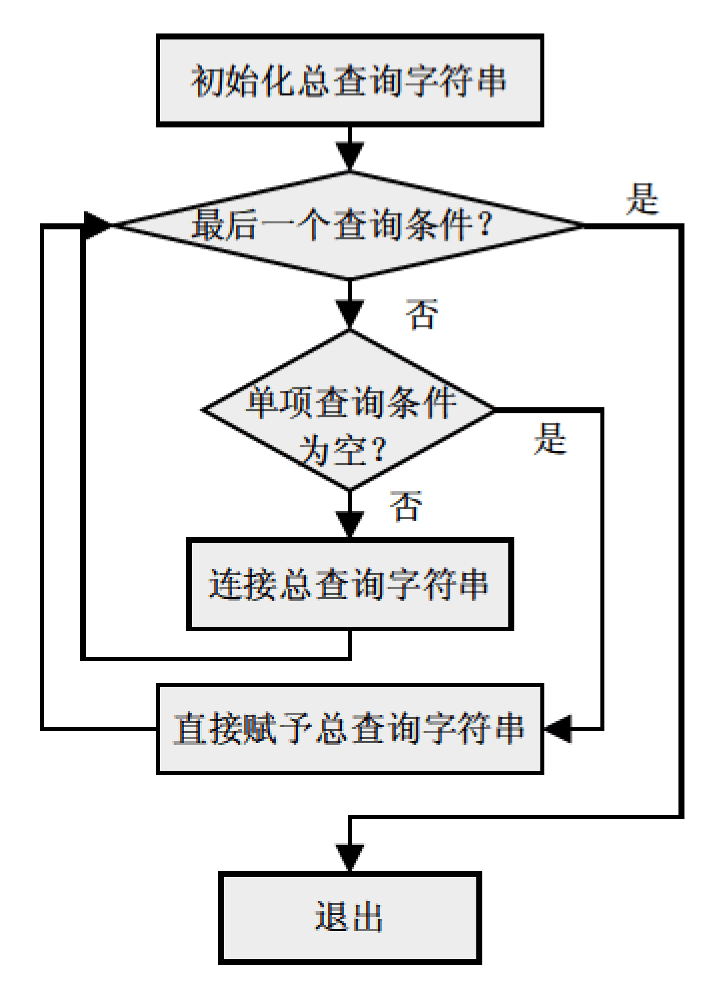


图 9-13　总查询字符串设置过程流程图

```
Sub SetSQL()                                    '修改总查询字符串
strSQL = ""                                     '初始化总查询字符串
'检查商场名查询条件是否为空，非空时，将该条件保存在查询字符串中
If Len(strMarketFilter) > 0 Then
    strSQL = strMarketFilter
End If
'检查品牌名查询条件是否为空
If Len(strMarkFilter) > 0 Then
    '当查询字符串有数据时，将字符串与品牌名查询条件连接，否则直接将该字符串赋给总查询字符串
    If Len(strSQL) > 0 Then
        strSQL = "(" & strSQL & ") and " & "(" & strMarkFilter & ")"
    Else
        strSQL = strMarkFilter
    End If
End If
'检查产品型号名查询条件是否为空
If Len(strProductFilter) > 0 Then
    '当查询字符串有数据时，将字符串与产品型号查询条件连接，否则直接将该字符串赋给总查询字符串
    If Len(strSQL) > 0 Then
        strSQL = "(" & strSQL & ") and " & "(" & strProductFilter & ")"
    Else
        strSQL = strProductFilter
    End If
End If
'检查产品尺寸查询条件是否为空
If Len(strSizeFilter) > 0 Then
    '当查询字符串有数据时，将字符串与产品尺寸查询条件连接，否则直接将该字符串赋给总查询字符串
    If Len(strSQL) > 0 Then
        strSQL = "(" & strSQL & ") and " & "(" & strSizeFilter & ")"
    Else
        strSQL = strSizeFilter
    End If
End If
'检查销售数量查询条件是否为空
If Len(strNumberFilter) > 0 Then
    '当查询字符串有数据时，将字符串与销售数量查询条件连接，否则直接将该字符串赋给总查询字符串
    If Len(strSQL) > 0 Then
        strSQL = "(" & strSQL & ") and " & "(" & strNumberFilter & ")"
    Else
        strSQL = strNumberFilter
    End If
End If
'检查销售单价查询条件是否为空
If Len(strPriceFilter) > 0 Then
    '当查询字符串有数据时，将字符串与销售单价查询条件连接，否则直接将该字符串赋给总查询字符串
    If Len(strSQL) > 0 Then
        strSQL = "(" & strSQL & ") and " & "(" & strPriceFilter & ")"
    Else
        strSQL = strPriceFilter
    End If
```

```
End If
'检查销售日期查询条件是否为空
If Len(strDateFilter) > 0 Then
    '当查询字符串有数据时，将字符串与销售日期查询条件连接，否则直接将该字符串赋给总查询字符串
    If Len(strSQL) > 0 Then
        strSQL = "(" & strSQL & ") and " & "(" & strDateFilter & ")"
    Else
        strSQL = strDateFilter
    End If
End If
End Sub
```

代码说明：

- 从流程图中可以看到，需要对每一个单项查询条件都进行检测，查看这些查询条件是否为空，从而保证最终的总查询字符串得到正确的结果。在该段代码中没有对各个单项查询条件写入数组，因而只能使用逐个检测的方式实现。读者可以试着通过数组的方式循环检测来实现同样的功能。
- 最终得到的总查询条件的格式应该是：(某单个查询条件)+(某单个查询条件)+……。其中的括号是在单项查询条件与总查询条件都非空情况被添加进去的。当总查询条件为空的时候，仅仅出现了一个单个查询条件，括号的设置没有必要，因此这种情况下舍去了括号。
- 实际上在该过程中产生的总查询条件字符串还不是完成的 SQL 查询语句，这里得到的仅仅是其中的条件部分，因此不能直接使用该总查询条件字符串完成 SQL 查询。

9.3.4　数据库建立与更新过程代码设计

前面在数据表设计章节提到商品销售数据资料表的结构不是通过手动设计的，而是通过代码设计。本小节将解释该段代码，该段代码将生成商品销售数据资料表的结构，并且它还完成该表的添加、修改记录操作。如图 9-14 所示的是该过程的流程图。

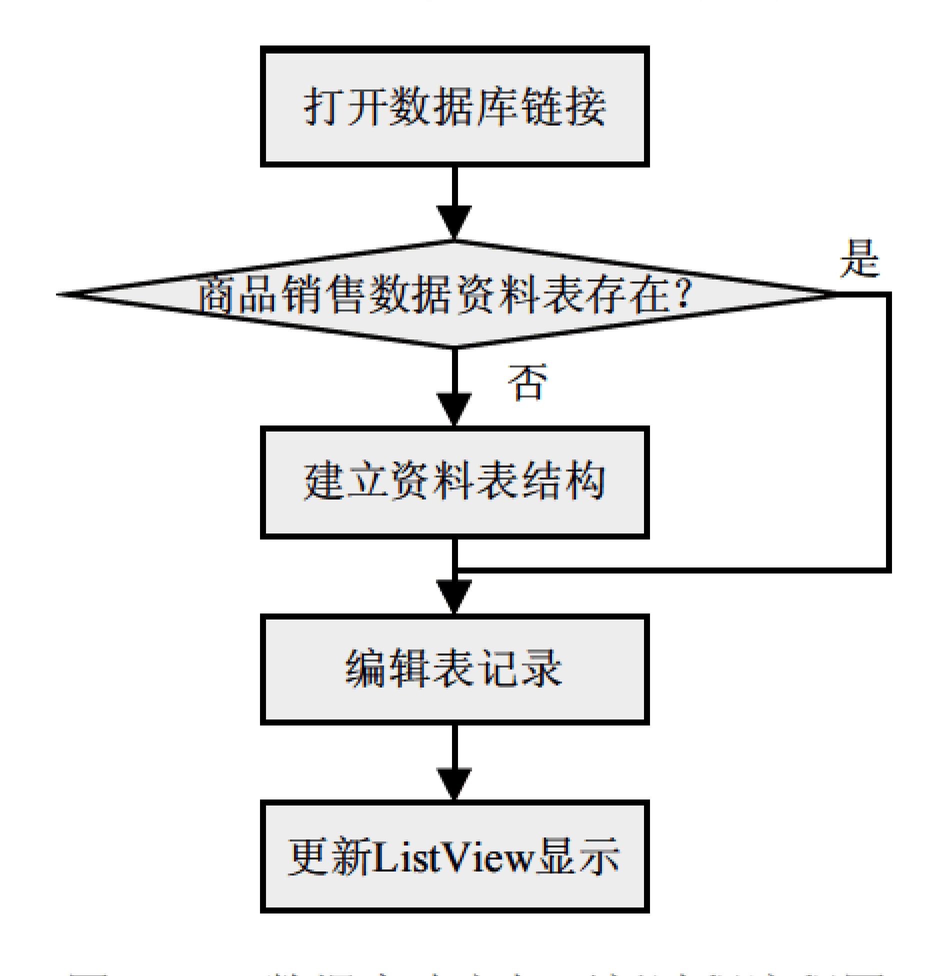


图 9-14　数据库建立与更新过程流程图

在上面的流程图中，编辑表记录过程如何被执行将会由传递到总过程的参数决定。因为对记录的编辑工作包含了新建、修改、删除操作，因而该过程共包含了对应的 3 个部分。

```
'建立数据库与数据表，向数据库的数据表中输入数据
Sub RefreshDB(intOPMethod As Integer, _
                Optional strMarket As String, _
                Optional strMark As String, _
                Optional strProduct As String, _
                Optional strSize As String, _
                Optional strNumber As String, _
                Optional strPrice As String)
Dim tdf As DAO.TableDef, fld As DAO.Field, IsFind As Boolean, rs As DAO.Recordset, db As
DAO.Database
Dim strSQL As String
'打开数据库链接
Set db = OpenDataBase(ThisWorkbook.Path & "\DB\DB.mdb")
'循环数据库中所有的表对象，检测是否由表 t_Sale 存在
isFind=False
For Each tdf In db.TableDefs
    If tdf.Name="t_Sale" Then
        IsFind = True
        Exit For
    End If
Next
'当没有找到表 t_Sale 时，在数据库中建立该表
If Not IsFind Then
    Set tdf = db.CreateTableDef("t_Sale")                    '建立表 t_Sale
    Set fld = tdf.CreateField("MarketName", dbText, 50)      '设置商场名称字段
    tdf.Fields.Append fld                                    '将商品名称字段添加到表 t_Sale 中
    Set fld = tdf.CreateField("Mark", dbText, 50)            '设置品牌字段
    tdf.Fields.Append fld                                    '将品牌名称字段添加到表 t_Sale 中
    Set fld = tdf.CreateField("Product", dbText, 50)         '设置产品型号字段
    tdf.Fields.Append fld                                    '将产品型号字段添加到表 t_Sale 中
    Set fld = tdf.CreateField("Size", dbText, 50)            '设置产品尺寸字段
    tdf.Fields.Append fld                                    '将产品尺寸字段添加到表 t_Sale 中
    Set fld = tdf.CreateField("Number", dbInteger)           '设置销售数量字段
    tdf.Fields.Append fld                                    '将销售数量字段添加到表 t_Sale 中
    Set fld = tdf.CreateField("Price", dbDouble)             '设置销售单价字段
    tdf.Fields.Append fld                                    '将销售单价字段添加到表 t_Sale 中
    Set fld = tdf.CreateField("Date", dbInteger)             '设置销售日期字段
    tdf.Fields.Append fld                                    '将销售日期字段添加到表 t_Sale 中
    Set fld = tdf.CreateField("ID", dbText, 14)              '设置 ID 字段
    tdf.Fields.Append fld                                    '将 ID 字段添加到表 t_Sale 中
    db.TableDefs.Append tdf                                  '将表 t_Sale 添加到数据库中
End If
'对商品销售数据表记录进行编辑
'intOPMethod---1 代表新建记录，2 代表修改记录，3 代表删除记录
Select Case intOPMethod
    '新建记录，新建不需要更新查询结果窗口的数据显示，因为新建只在销售商品时发生
```

```
Case 1
    Set rs = tdf.OpenRecordset                              '打开记录集
    With rs
        .AddNew                                             '添加新记录
        .Fields("MarketName") = strMarket                   '获取商场名称字段
        .Fields("Mark") = strMark                           '获取品牌名称字段
        .Fields("Product") = strProduct                     '获取产品型号字段
        .Fields("Size") = strSize                           '获取产品尺寸字段
        .Fields("Number") = strNumber                       '获取产品数量字段
        .Fields("Price") = strPrice                         '获取产品单价字段
        .Fields("Date") = Format(Now, "DD")                 '获取销售时间字段
        .Fields("ID") = Format(Now, "YYYYMMDDHHMMSS")       '获取记录 ID 字段
        .Update                                             '更新记录集数据
    End With
'修改记录
Case 2
    strSQL = "select * from t_Sale where ID=" & arrEdited(7)    '获取查询字符串
    Set rs = db.OpenRecordset(strSQL)                       '获取对应 ID 的记录
    With rs
        .MoveFirst
        .Edit                                               '开始编辑记录
        .Fields("MarketName") = strMarket                   '修改商场名称字段
        .Fields("Mark") = strMark                           '修改品牌名称字段
        .Fields("Product") = strProduct                     '修改产品型号字段
        .Fields("Size") = strSize                           '修改产品尺寸字段
        .Fields("Number") = strNumber                       '修改产品数量字段
        .Fields("Price") = strPrice                         '修改产品单价字段
        .Fields("Date") = Format(Now, "DD")                 '修改销售时间字段
        .Fields("ID") = arrEdited(7)                        '修改记录 ID 字段
        .Update                                             '更新记录集数据
    End With
    '在修改记录时，可能用户是修改的在查询结果中选择了部分结果后的记录。这些记录被程序存
     储在临时表中，此时对于表 t_Sale 和临时表的数据都要进行更新，下面代码是更新临时表数据
    If Not isSaveAll Then
        strSQL = "Select * from TempTable" & " where ID=" & arrEdited(7)
        Set rs = db.OpenRecordset(strSQL)                   '打开临时表记录集
        '开始修改临时记录集，修改步骤和前面修改 t_Sale 类似
        With rs
            .MoveFirst                                      '移动记录集指针到第一条记录
            .Edit                                           '开始编辑记录集
            .Fields("MarketName") = strMarket               '指定记录 MarketName 字段的值
            .Fields("Mark") = strMark                       '指定记录 Mark 字段的值
            .Fields("Product") = strProduct                 '指定记录 Product 字段的值
            .Fields("Size") = strSize                       '指定记录 Size 字段的值
            .Fields("Number") = strNumber                   '指定记录 Number 字段的值
            .Fields("Price") = strPrice                     '指定记录 Price 字段的值
            .Fields("Date") = Format(Now, "DD")             '指定记录 Date 字段的值
            .Fields("ID") = arrEdited(7)                    '指定记录 ID 字段的值
            .Update                                         '更新记录集
```

```
            End With
        End If
        Set rs = Nothing
        '以下代码是更新查询结果窗体中 ListView 控件的显示
        '在修改记录时，只需要更新在 ListView 控件中被选择的记录
        With frmQryResult.ListViewQry.SelectedItem
            .Text = strMarket
            .SubItems(1) = strMark                    '设置被选定修改项目的第一个子项目
            .SubItems(2) = strProduct                 '设置被选定修改项目的第二个子项目
            .SubItems(3) = strSize                    '设置被选定修改项目的第三个子项目
            .SubItems(4) = strNumber                  '设置被选定修改项目的第四个子项目
            .SubItems(5) = strPrice                   '设置被选定修改项目的第五个子项目
            .SubItems(6) = Format(Now, "D")           '设置被选定修改项目的第六个子项目
            .SubItems(7) = arrEdited(7)               '设置被选定修改项目的第七个子项目
        End With
    Case 3
        strSQL = "delete * from t_Sale where ID=" & arrEdited(7)     '设置删除查询字符串
        db.Execute (strSQL)                                          '删除记录
        '删除在临时表中的对应记录
        If Not isSaveAll Then
            db.Execute ("delete * from TempDB" & " where ID=" & arrEdited(7))
        End If
        '根据具体情况获取修改后的记录集
        If isSaveAll Then
            Set rs = db.OpenRecordset("select * from t_Sale")        '打开到表 t_Sale 的记录集
        Else
            Set rs = db.OpenRecordset("Select * from TempTable")     '打开到表 TempTable 的记录集
        End If
        '重置 ListView 控件内部显示项目
        frmQryResult.ListViewQry.ListItems.Clear                     '清除控件所有项目
        Do Until rs.EOF
            '为控件添加一个新项目
            Set itemList = frmQryResult.ListViewQry.ListItems.Add(Text:=rs.Fields("MarketName"))
            With itemList
                .SubItems(1) = rs.Fields("Mark")                     '设置控件第一个子项目
                .SubItems(2) = rs.Fields("Product")                  '设置控件第二个子项目
                .SubItems(3) = rs.Fields("Size")                     '设置控件第三个子项目
                .SubItems(4) = rs.Fields("Number")                   '设置控件第四个子项目
                .SubItems(5) = rs.Fields("Price")                    '设置控件第五个子项目
                .SubItems(6) = rs.Fields("Date")                     '设置控件第六个子项目
                .SubItems(7) = rs.Fields("ID")                       '设置控件第七个子项目
                rsCount = rsCount + 1                                '记忆记录位置
                rs.MoveNext                                          '移动到记录集下一条记录
            End With
        Loop
End Select
'清除临时变量占用空间
Set rs = Nothing
Set tdf = Nothing
```

```
Set fld = Nothing
Set db = Nothing
End Sub
```

代码说明：

- 在使用 DAO 向数据库添加新表时，首先需要使用数据库对象的 CreateTableDef 方法建立一个表对象，然后使用该表对象的 CreateField 方法为表建立字段。在字段信息建立后，需要使用表对象的 Fields 集合的 Append 方法将字段添加到表的字段集中。所有的字段都添加完成后，需要使用数据库对象的 tableDefs 表集合的 Append 方法将表添加到数据库的表集合中。
- 编辑记录时有 3 种情况，分别是新建、修改和删除。新建操作只在商品销售时出现，它不需要完成后续的更新 ListView 控件显示操作。在修改和删除记录时，还需要考虑一种情况。系统中允许用户将查询结果中的数据选择部分后单独显示，并且将该部分数据单独导出。该功能是通过建立临时表实现的，而当修改和删除属于此种情况下的数据时，修改数据库中的数据时需要同时修改表 t_Sale 和临时表。

9.3.5 压缩数据库代码设计

数据库文件在系统使用一段时间后，其大小会急剧变大，这不完全是由于数据增多造成的。DAO 提供了压缩数据库的功能，此功能可以极大地减少数据冗余，并且可以加快数据库的访问速度。以下是该过程的代码：

```
'压缩数据库
Public Sub ZipDB()
On Error GoTo Exit_sub
'压缩商品销售登记资料表
DAO.DBEngine.CompactDatabase ThisWorkbook.Path & "\DB\DB.mdb", _
    ThisWorkbook.Path & "\DB\tempDB.mdb", dbLangChineseSimplified
Kill ThisWorkbook.Path & "\DB\DB.mdb"
FileCopy ThisWorkbook.Path & "\DB\tempDB.mdb", ThisWorkbook.Path & "\DB\DB.mdb"
Kill ThisWorkbook.Path & "\DB\tempDB.mdb"
'压缩基本信息资料表
DAO.DBEngine.CompactDatabase ThisWorkbook.Path & "\DB\Info.mdb", _
    ThisWorkbook.Path & "\DB\tempInfo.mdb", dbLangChineseSimplified
Kill ThisWorkbook.Path & "\DB\Info.mdb"
FileCopy ThisWorkbook.Path & "\DB\tempInfo.mdb", ThisWorkbook.Path & "\DB\Info.mdb"
Kill ThisWorkbook.Path & "\DB\tempInfo.mdb"
MsgBox "数据库文件压缩操作成功！ ", vbOKOnly + vbInformation, "压缩数据库"
Exit Sub
Exit_sub:
    MsgBox "请检查数据库文件 DB.mdb 和 Info.mdb 是否存在！", vbOKOnly + vbInformation, "数据库
文件丢失"
End Sub
```

代码说明：

- ❑ 通过 CompatDatabase 方法压缩数据库时，它将原数据文件压缩后保存为一个新文件。为了保证这些数据库文件的名称在压缩前后不发生变化，需要将原数据库文件删除，然后修改新数据库文件为原数据库文件的名称。
- ❑ CompatDatabase 不能在数据库文件被打开状态下使用。在使用该方法时，应确保没有开启需压缩的数据库文件。

9.4 基本信息设置窗体设计

窗体设计是本章的重点。在前面设计思路小节中已经对本实例中所涉及到的窗体做了大致介绍，随后的各节将详细展开。对于实例中所有的窗体，都将分界面设计与代码设计介绍，以便于理解。由于每个窗体的代码都比较多，因此，本章将对每一个窗体都分一个章节介绍，并且代码介绍都细化到单个控件。

9.4.1 基本信息设置窗体界面设计

基本信息设置窗体用于建立基本信息，这些资料将被写入基本信息资料表的对应表中。需要建立的基本资料包括商场名称、品牌名称以及商品名称。商品的名称中包含了尺寸，所以没有建立相应的尺寸项。该窗体的界面如图 9-15 所示。

表 9-1 列出了该窗体中使用到的控件的控件名、控件类型以及用途的说明。

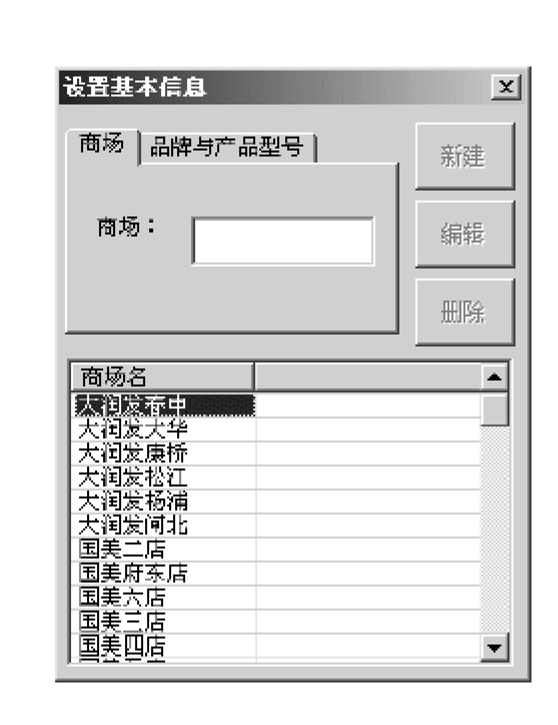


图 9-15 基本信息设置窗体界面

表 9-1 基本信息设置窗体控件列表

控 件 名	控 件 类 型	说 明
MultiPage1	多页控件	控件包含了两个页面，在各个页面中，分别对商场数据和品牌、产品型号数据进行编辑
ListView	ListView 控件	控件显示已经建立的数据。当在多页控件中选择商场时，该多页控件显示的是已建立的商场资料。当选择品牌与产品型号时，该控件显示相应的已建立数据
btnNew	按钮控件	单击该按钮，将把对应多页控件中的文本框数据写入资料表中。选择的多页控件不同，写入的表也不同
btnEdit	按钮控件	只有在 ListView 控件中选择了项目，该按钮才会被激活。单击该按钮后，将对选择项目的数据进行修改
btnDelete	按钮控件	单击该按钮后，将把选择的项目从资料表中删除

建立该窗体的步骤如下：

（1）在 Excel 2007 的 VBE 开发环境中依次选择【插入】|【用户窗体】命令插入一个新窗体，如图 9-16 所示。在新插入窗体的属性窗口中设置名称属性为 frmSetup，如图 9-17 所示。



图 9-16　插入用户窗体



图 9-17　修改窗体名称

（2）在工具箱中选择多页控件，在窗体上单击鼠标左键并拖动创建一个新多页控件。默认的多页控件已经建立了两个页。随后在属性窗口中将第一页的名称属性设置为 pMarket，Caption 设置为“商场”。将第二页的名称属性设置为 pMarkProduct，Caption 属性设置为“品牌与产品型号”。

（3）选择【商场】页标签，在该页面里插入一个标签控件和一个文本框控件。随后在属性窗口中将标签控件的 Caption 属性设置为“商场：”。文本框控件的名称属性设置为 txtMarket，其 SelectionMargin 属性设置为 False。最后通过鼠标调整好控件在该页面上的位置。最终效果如图 9-18 所示。

（4）选择【品牌与产品型号】页标签，在该页面里插入两个标签控件和两个文本框控件。在属性窗口中将第一个标签的 Caption 属性设置为“品牌：”，第二个标签的 Caption 属性设置为“产品型号：”。设置第一个文本框控件的名称属性为 txtMark，第二个文本框控件的名称属性为 txtProduct。然后将两个文本框控件的 SelectionMargin 属性都设置为 False。最后调整好控件在该页的位置，最终效果如图 9-19 所示。

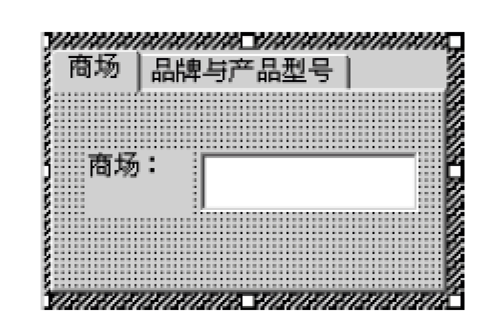


图 9-18　商场页面控件调整

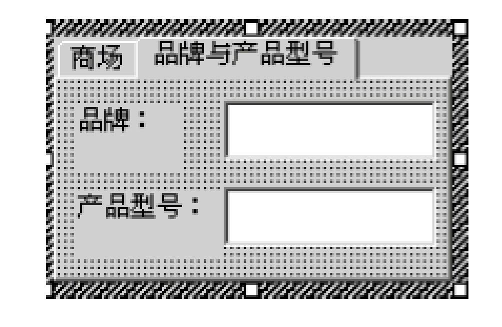


图 9-19　品牌与产品型号页面控件调整

（5）在工具箱中选择按钮控件，然后在窗体中连续插入 3 个按钮控件。在属性窗口中将 3 个按钮的名称属性依次设置为 btnNew、btnEdit 和 btnDelete。最后调整好 3 个按钮在窗体上的位置。

（6）在工具箱中找到 ListView 控件并在窗体中插入一个 ListView 控件。在属性窗口中设置该控件的名称属性为“ListView”。最后调整好该控件大小即可。最终效果如图 9-15 所示。

在工具箱中没有找到 ListView 控件时，需要建立引用。方法是：在工具菜单中选择【引用】命令，在打开的【引用】对话框中选择 Microsoft Windows Common Controls 6.0 项目，如图 9-20 所示。若在左侧的列表框中没有找到该项目，可以选择【浏览】，按照所示定位中显示的地址找到该文件即可。然后在工具箱空白区域右击，在弹出的快捷菜单中选择【附加控件】命令，再在打开的【附加控件】中选择 Microsoft ListView Control, version 6.0 项目并单击

【确定】按钮，如图 9-21 所示。在工具箱中就会出现 ListView 控件的快捷插入按钮。

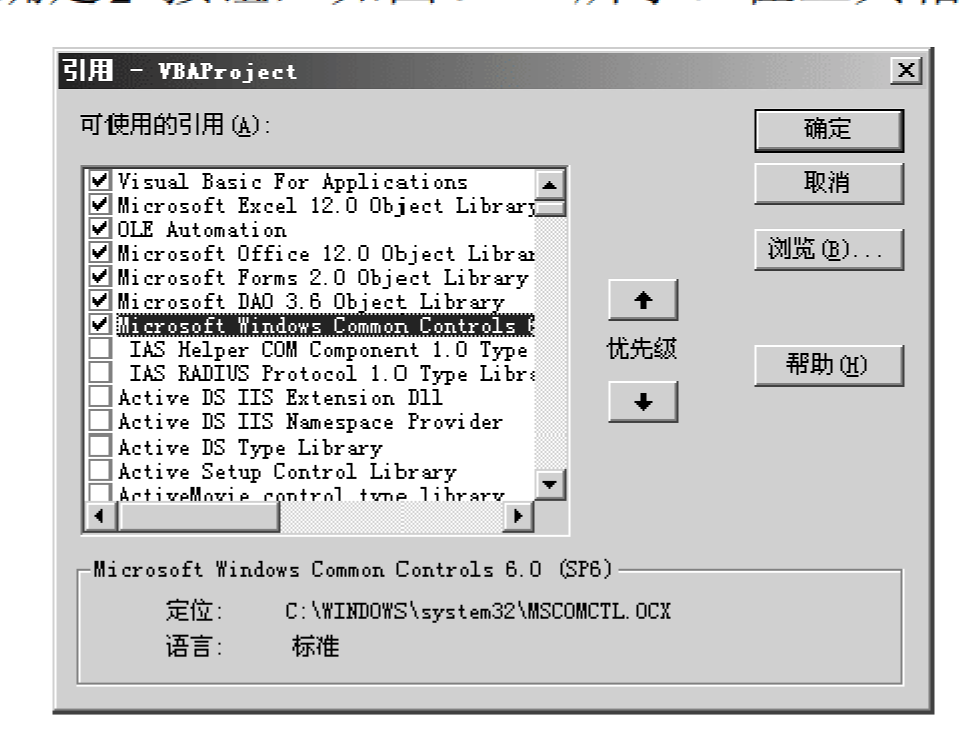


图 9-20 【引用】对话框

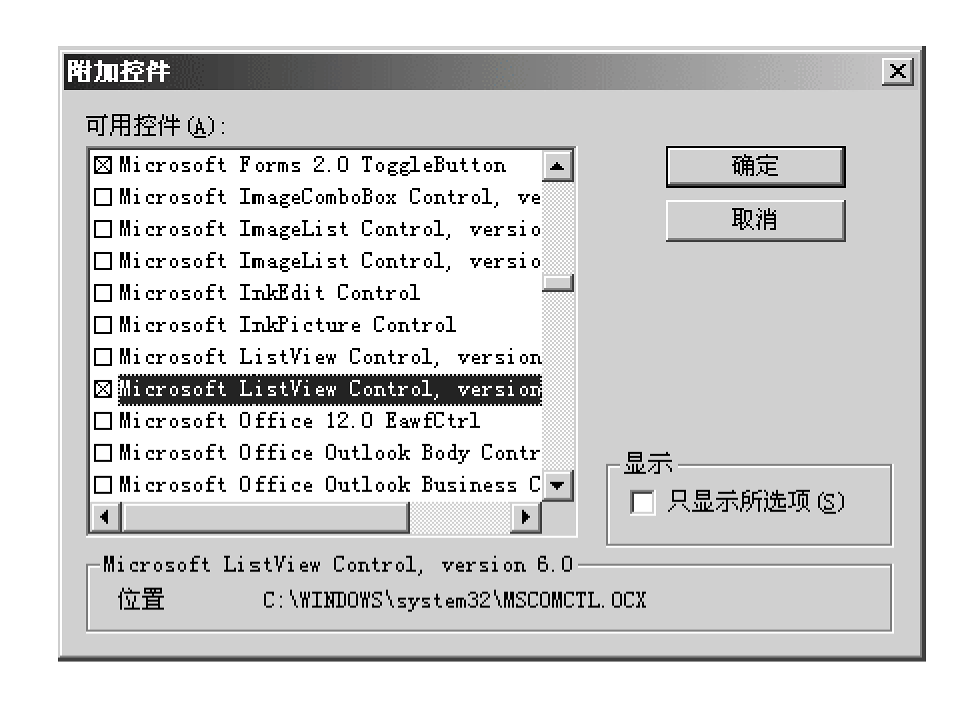


图 9-21 【附加控件】对话框

9.4.2 窗体初始化代码

基本信息设置窗体需要初始化的内容不多。当窗体被打开后，多页控件默认选择的是商场页面。商场页面需要激活，另外 ListView 控件也要显示已建立的所有商场信息。3 个按钮的状态也需要设置，这些工作都可以在窗体初始化过程中找到。以下是窗体的初始化代码：

```
Private Sub UserForm_Initialize()
'激活商场页面
MultiPage1.Value = 0
'代码触发商场页面单击事件，完成 ListView 列表内容初始化以及按钮状态初始化
MultiPage1_Click 0
'单击 ListView 控件第一列时不变成编辑状态
ListView.LabelEdit = lvwManual
'ListView 控件失去焦点时仍能显示选中的项目
ListView.HideSelection = False
End Sub
```

代码说明：

- ❑ 多页控件的页面单击事件是初始化过程中的一个主体。该事件完成了 ListView 控件状态与内容初始化以及按钮的状态初始化工作。该过程的代码在本节后续部分进行介绍。
- ❑ ListView 控件的 LabelEdit 属性可以控制单击第一列的项目时，是否进入项目的编辑状态。该属性的默认值为 lvwAutomatic，设置该值将会自动进入编辑状态。
- ❑ ListView 控件的 HideSelection 属性可以控制当 ListView 控件失去焦点时，控件中原来被选中的项目是否仍然处于被选中状态。设置为 False 时，控件失去焦点仍能显示选中的项目。

多页控件的页面被单击时将触发多页控件页面单击事件，该事件还传递了一个 Index 参数，该参数指定了页面的索引。该页面单击事件的代码很长，在阅读之前，首先应了解其工作流程。该过程的流程图如图 9-22 所示。

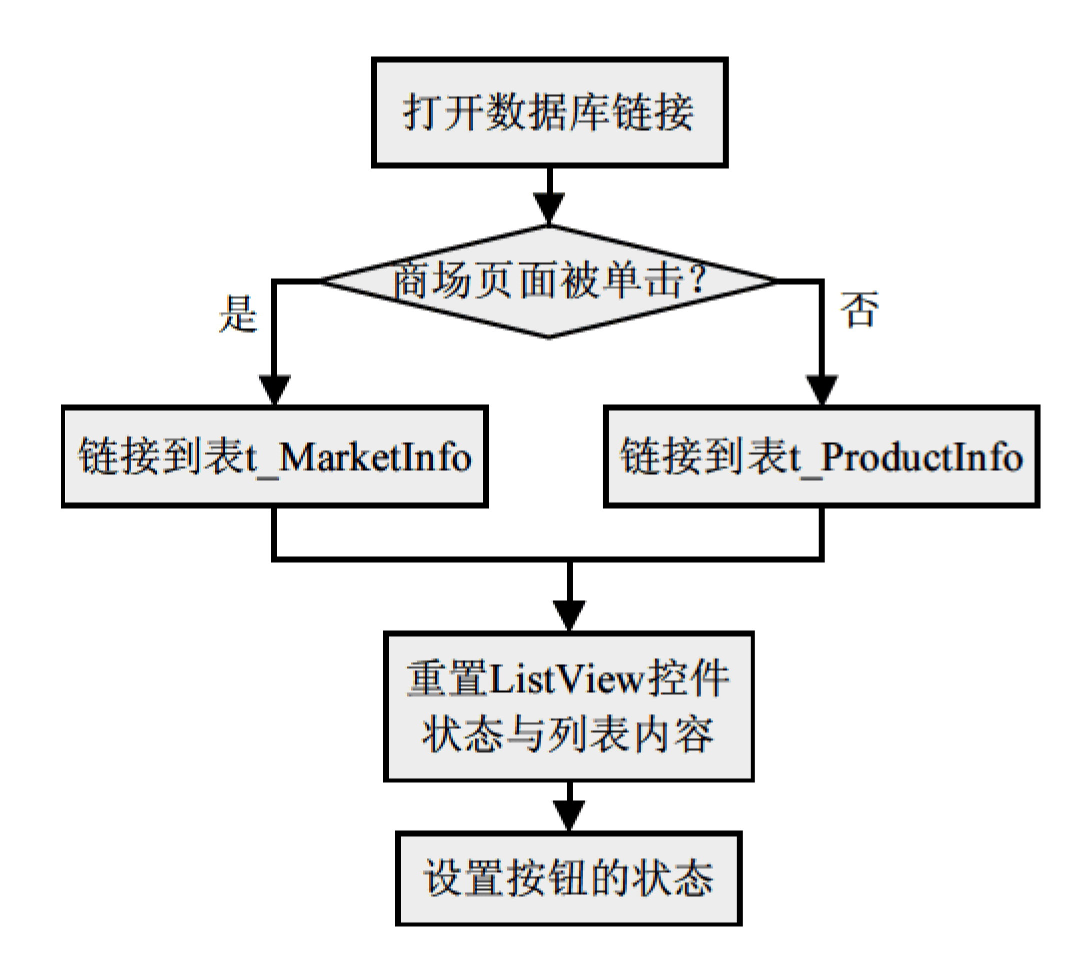


图 9-22　多页控件单击事件过程流程图

需要说明的是：在图 9-22 中，重置 ListView 控件状态与列表内容与设置按钮的状态两个步骤。对于单击不同页面时其完成的实际工作是不一样的，因为它们都需要针对不同页面，使用链接到的不同表的数据。以下是该单击事件过程的代码：

```
Private Sub MultiPage1_Click(ByVal Index As Long)
Dim db As DAO.Database, rs As DAO.Recordset
Dim itemList As ListItem
'打开数据库链接
Set db = OpenDataBase(ThisWorkbook.Path & "\DB\Info.mdb")
'根据页面索引完成各自的初始化工作，商场页面的索引为 0
If Index Then
    '初始化 ListView 控件状态
    With ListView
        .Gridlines = True                       '显示网格线
        .FullRowSelect = True                   '允许整行选择
        .MultiSelect = False                    '不允许多行选择
        .LabelEdit = lvwManual                  '单击 ListView 控件第一列时不变成编辑状态
        .View = lvwReport                       'ListView 显示模式为报告模式
        '设置 ListView 控件标题
        With .ColumnHeaders
            .Clear
            .Add Text:="品牌", Width:=74        '第一列列头显示为品牌，宽度为 74
            .Add Text:="产品型号", Width:=74    '第二列列头显示为产品型号，宽度为 74
        End With
    End With
    '获取连接到 t_ProductInfo 表的记录集，并且该记录集按照品牌名进行排序
    Set rs = db.OpenRecordset("select * from t_ProductInfo order by Mark")
    '清空 ListView 控件列表内容
    ListView.ListItems.Clear
    '将记录集中所有项目写入 ListView 控件中
    Do Until rs.EOF
        '为 ListView 控件添加新项，该项使用记录集的品牌名建立，显示在 ListView 控件第一列
        Set itemList = ListView.ListItems.Add(Text:=rs.Fields("Mark"))
```

```
            '为新项添加子项，该子项使用记录集的产品名称建立，显示在 ListView 控件第二列
            itemList.SubItems(1) = rs.Fields("Product")
            rs.MoveNext                                 '将记录集的指针移到下一条记录
        Loop
        '设置按钮的可用状态
        If Len(Trim(txtMark.Text)) And Len(Trim(txtProduct.Text)) Then
            '当品牌名称与产品名称文本框都输入内容时，所有按钮可用，否则都不可用
            btnNew.Enabled = True                       '新建按钮可用
            btnEdit.Enabled = True                      '编辑按钮可用
            btnDelete.Enabled = True                    '删除按钮可用
        Else
            btnNew.Enabled = False                      '新建按钮不可用
            btnEdit.Enabled = False                     '编辑按钮不可用
            btnDelete.Enabled = False                   '删除按钮不可用
        End If
    Else
        '初始化 ListView 控件状态
        With ListView
            .Gridlines = True                           '显示网格线
            .FullRowSelect = True                       '允许整行选择
            .MultiSelect = False                        '不允许多行选择
            .LabelEdit = lvwManual                      '单击 ListView 控件第一列时不变成编辑状态
            .View = lvwReport                           'ListView 显示模式为报告模式
            '设置 ListView 控件标题
            With .ColumnHeaders
                .Clear
                .Add Text:="商场名", Width:=74          '第一列列头显示为商场名，宽度为 74
            End With
        End With
        '获取连接到 t_MarketInfo 表的记录集，并且该记录集按照商场名进行排序
        Set rs = db.OpenRecordset("select * from t_MarketInfo order by MarketName")
        '清空 ListView 控件列表内容
        ListView.ListItems.Clear
        '将记录集中所有项目写入 ListView 控件中
        Do Until rs.EOF
            '为 ListView 控件添加新项，该项使用记录集的商场名建立
            Set itemList = ListView.ListItems.Add(Text:=rs.Fields("MarketName"))
            rs.MoveNext                '将记录集的指针移到下一条记录
        Loop
        '设置按钮的可用状态
        btnNew.Enabled = Len(Trim(txtMarket.Text))          '设置新建按钮可用状态
        btnEdit.Enabled = Len(Trim(txtMarket.Text))         '设置编辑按钮可用状态
        btnDelete.Enabled = Len(Trim(txtMarket.Text))       '设置删除按钮可用状态
    End If
    '清除临时对象变量占用的内存空间
    Set itemList = Nothing
    Set rs = Nothing
    Set db = Nothing
End Sub
```

代码说明：

- 要使用数据库文件中的表建立记录集，首先需要链接到数据库。通过 DAO 对象的 OpenDataBase 函数可以完成该工作，函数获取数据库文件的位置信息即可获取数据库对象。
- 程序中记录集的获取是通过使用 SQL 语句实现的，其中的 Order By 关键字指定了记录集将按照哪个字段进行排序。
- 要为 ListView 控件添加标题时，应该使用该控件的 ColumnHeaders 集合对象，然后使用该对象的 Add 方法建立新的标题。这些标题的排列顺序将按照代码的先后顺序出现，要清空标题只需要使用该对象的 Clear 方法。
- 要为 ListView 控件添加项目时，需要使用该对象的 ListItems 集合对象，然后使用该对象的 Add 方法添加新项目。该新项目建立时，只能指定第一列的内容。要为新项目添加其他列的内容，需要使用该新项目 SubItems 集合来依次指定。

9.4.3　新建按钮代码设计

窗体中的【新建】按钮对于选择的不同页面将会采取不同的操作。当用户选择的是商场页面时，【新建】按钮将链接商场销售数据资料表，为该表添加新记录；当用户选择的是品牌与产品型号页面时，【新建】按钮将链接商品信息资料表，并向该表添加新记录。如图 9-23 所示的是该【新建】按钮的运行流程图。

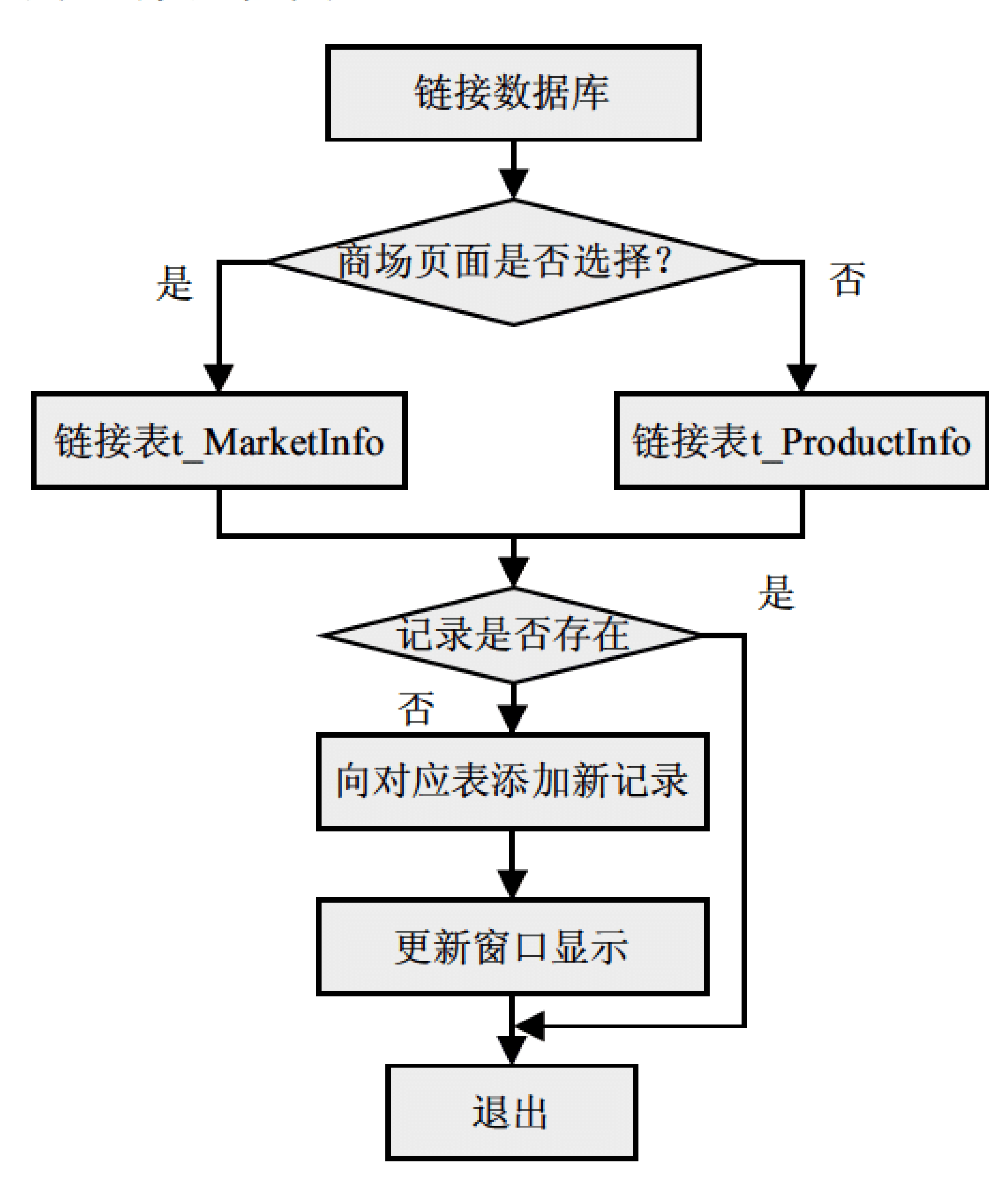


图 9-23　【新建】按钮运行流程图

```
Private Sub btnNew_Click()
Dim db As DAO.Database, rs As DAO.Recordset
'链接数据库
```

```
Set db = OpenDataBase(ThisWorkbook.Path & "\DB\Info.mdb")
'根据不同情况分别完成添加新记录与更新窗口显示工作
If MultiPage1.SelectedItem.Index Then
     '当用户选择品牌与产品型号页面时，从商品信息表中获取满足条件的记录集
     '这个条件是指品牌与产品型号与页面文本框中输入数据相等
    Set rs = db.OpenRecordset("select * from t_ProductInfo where Mark='" & txtMark.Text & "' and
Product='" & txtProduct.Text & "' order by Mark")
     '检查获得的记录集是否有记录
    If rs.RecordCount > 0 Then
        MsgBox "该项已经存在！ ", vbOKOnly + vbInformation
    Else
        '当输入的信息资料在商品资料表中不存在时，将新资料添加到表中
        rs.AddNew                                          '添加新记录
        rs.Fields("Mark") = txtMark.Text                   '将品牌信息写入表中的品牌字段
        rs.Fields("Product") = txtProduct.Text             '将商品名称写入表中的商品名称字段
        '以下代码将根据商品名称获取该产品的尺寸信息，并将该信息写入商品信息表中
        '在产品型号最左端开始查询，首次出现的连续两个阿拉伯数字即为该产品尺寸
        i = 1
        Do Until i = Len(txtProduct.Text)
            j = Asc(Mid(txtProduct.Text, i, 1))            '获取产品型号第 i 个字符的 ASCII 码
            If j >= 48 And j <= 57 Then
                k = Asc(Mid(txtProduct.Text, i + 1, 1))    '当第 i 个字符为数字，检查第 i+1 个字符
                If k >= 48 And k <= 57 Then
                    rs.Fields("Size") = "S" & Mid(txtProduct.Text, i, 2)  '将该尺寸前面添加“S”后
                                                                           写入表中
                    Exit Do                                '找到尺寸后直接退出循环
                End If
            End If
            i = i + 1                                      '累加循环变量
        Loop
        rs.Update                                          '更新记录集
        MultiPage1_Click MultiPage1.SelectedItem.Index     '更新窗口显示
    End If
Else
    '从商场信息资料表中获取商场名称字段与商场名称文本框相等的记录集
    Set rs = db.OpenRecordset("select * from t_MarketInfo where MarketName='" & txtMarket.Text & "'")
    '检查该记录集是否为空，为空时，添加新记录
    If rs.RecordCount > 0 Then
        MsgBox "该项已经存在！ ", vbOKOnly + vbInformation
    Else
        rs.AddNew                                          '添加新记录
        rs.Fields("MarketName") = txtMarket.Text           '写入商场名称
        rs.Update                                          '更新记录集
        MultiPage1_Click MultiPage1.SelectedItem.Index     '更新窗口显示
    End If
End If
'清除临时对象占用的内存空间
Set rs = Nothing
Set db = Nothing
End Sub
```

代码说明：

- 在检查表中是否有相同记录存在时，是通过从表中获取满足指定条件的记录集做到的。这些记录集的部分字段应该和页面里文本框的内容一致。当该记录集有记录时即说明已经在表中建立该资料的信息，这个时候将获得提示信息，否则将进入写入操作。
- 该系统中商品的尺寸实际上已经包含在了商品型号中。商品型号中首次出现的连续两位阿拉伯数字就是该商品的尺寸。程序中通过一个 Do…Until 循环检测商品型号中所有字符的 ASCII 码。当发现有连续两个阿拉伯数字出现时，立即退出循环，从而获取正确的尺寸资料。
- 在使用 DAO 对象新建、修改活删除记录时，当完成了所有的操作后，务必使用 Update 更新该记录集。只有当使用了该方法后，该记录集所做的新建、修改和删除操作才能最终体现出来。
- 在添加了新的记录后，在 ListView 控件中将立即显示该新记录。这里的方法是通过再次调用该页面的单击事件完成的，该事件的详细代码已经在前面做过介绍。它将会重新设置 ListView 控件的状态和内容以及各个按钮的状态。

9.4.4　编辑按钮代码设计

单击【编辑】按钮后，程序完成的操作大体上和【新建】按钮一致，它也需要对记录进行存在性检测，不过这里是要对修改后的记录进行存在性检测。【编辑】按钮单击事件过程的流程图如图 9-24 所示。

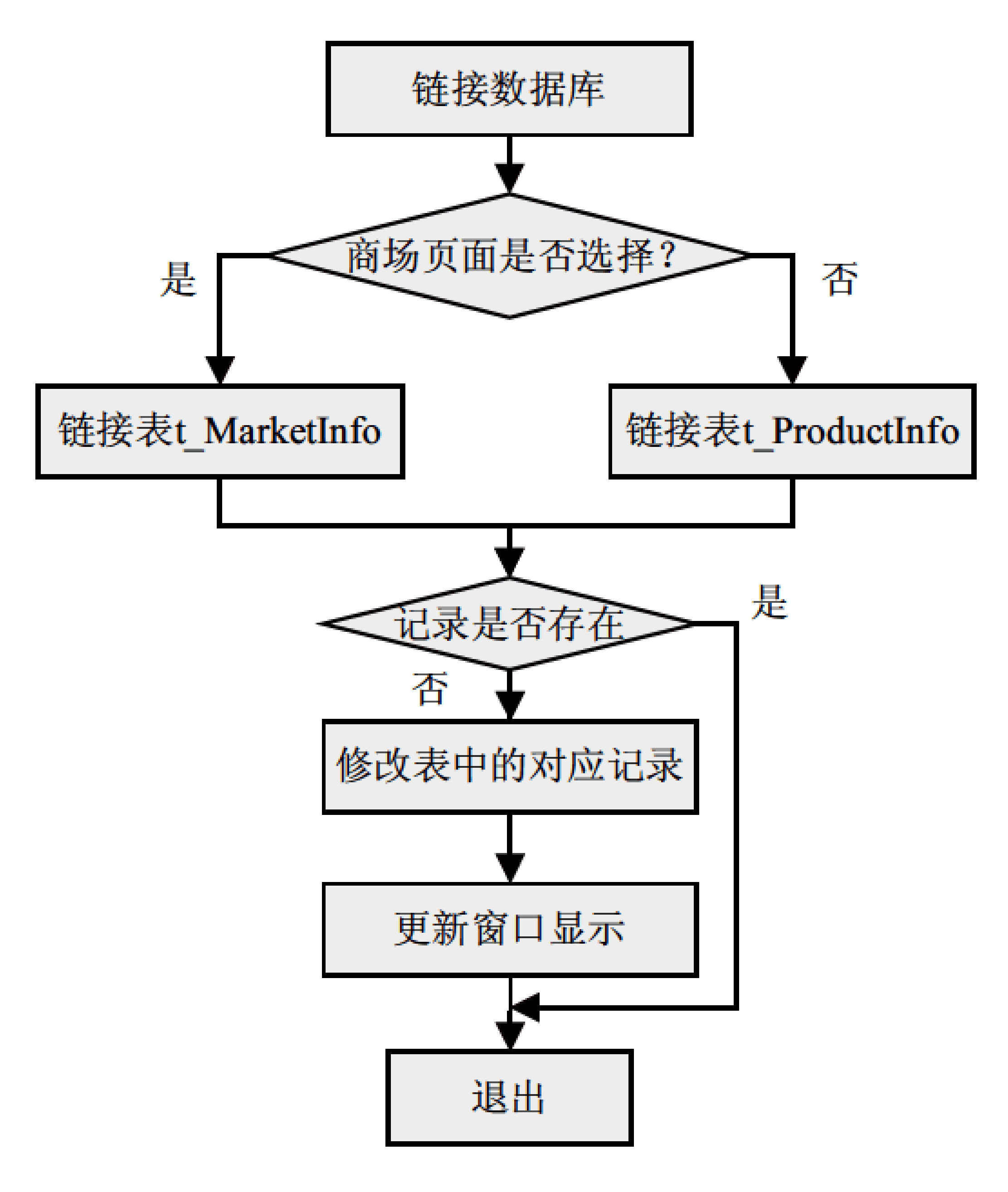


图 9-24　【编辑】按钮单击事件过程流程图

从图 9-24 中可以看出，该按钮的程序执行流程和【新建】按钮的执行流程相差不大。实际上该部分的代码都几乎一样。以下是该单击事件的程序代码：

```
Private Sub btnEdit_Click()
Dim db As DAO.Database, rs As DAO.Recordset
'链接数据库
Set db = OpenDataBase(ThisWorkbook.Path & "\DB\Info.mdb")
'根据不同情况分别完成记录编辑与更新窗口显示工作
If MultiPage1.SelectedItem.Index Then
    '当用户选择品牌与产品型号页面时，从商品信息表中获取满足条件的记录集
    '这个条件是指品牌与产品型号与页面文本框中输入数据相等
    Set rs = db.OpenRecordset("select * from t_ProductInfo where Mark='" & txtMark.Text & "' and
Product='" & txtProduct.Text & "'")
    '检查获得的记录集是否有记录
    If rs.RecordCount > 0 Then
        MsgBox "该修改结果已经存在！", vbOKOnly + vbInformation
        Exit Sub
    End If
    '打开满足原选择项目条件的记录集，然后修改该记录集中的数据
Set rs = db.OpenRecordset("select * from t_ProductInfo where Mark='" & ListView.SelectedItem.Text
& "' and Product='" & ListView.SelectedItem.SubItems(1) & "' order by Mark")
    rs.Edit                                        '开启记录集的编辑模式
    rs.Fields("Mark") = txtMark.Text               '修改品牌名称
    rs.Fields("Product") = txtProduct.Text         '修改产品型号
    '以下代码将获取产品尺寸
    i = 1                                          '初始化循环变量
    Do Until i = Len(txtProduct.Text)              '逐个检测产品型号字符串的每一个字符
        j = Asc(Mid(txtProduct.Text, i, 1))        '获取第 i 个字符的 ASCII 码
        If j >= 48 And j <= 57 Then                '检测第 i 个字符是否为数字
            k = Asc(Mid(txtProduct.Text, i + 1, 1))    '获取第 i+1 个字符的 ASCII 码
            If k >= 48 And k <= 57 Then            '检测第 i+1 个字符是否为数字
                rs.Fields("Size") = "S" & Mid(txtProduct.Text, i, 2)   '设置 Size 字段的值
                Exit Do
            End If
        End If
        i = i + 1                                          '循环变量加 1
    Loop
    rs.Update                                              '更新记录集
    MultiPage1_Click MultiPage1.SelectedItem.Index         '更新窗口显示
Else
    '从商场信息资料表中获取商场名称字段与商场名称文本框相等的记录集
    Set rs = db.OpenRecordset("select * from t_MarketInfo where MarketName='" & txtMarket.Text & "'")
    '检查该记录集是否为空，为空时，开始修改原记录
    If rs.RecordCount > 0 Then
        MsgBox "该项已经存在！", vbOKOnly + vbInformation
        Exit Sub
    End If
    '获取需要修改的记录的记录集
    Set rs = db.OpenRecordset("select * from t_MarketInfo where MarketName='" &
```

```
ListView.SelectedItem
.Text & "'")
      rs.Edit                                                 '开启记录集的编辑模式
      rs.Fields("MarketName") = txtMarket.Text                '修改商场名称
      rs.Update                                               '更新记录集
      MultiPage1_Click MultiPage1.SelectedItem.Index          '更新窗口显示
End If
'清除临时变量占用的内存资源
Set rs = Nothing
Set db = Nothing
End Sub
```

代码说明：

- 该段代码与【新建】按钮的代码有很多部分是相似的，这里不再加以说明。不同的地方是：当检测到修改后的数据在表中没有对应记录时，才开始编辑记录工作。要开始编辑记录，又需要重新获取新的记录集，该记录集满足的条件应该和 ListView 控件中选择项目的数据相同。此时的 SQL 语句使用了诸如 ListView.SelectedItem.Text 这样的表达式。
- 和记录集新建记录后需要使用 Update 更新记录集一样，编辑记录集完成也需要使用记录集的 Update 方法更新记录集。不然将会发生十分奇怪的事情：本来已经“完成”了新建、编辑工作，为何数据没有被建立、修改过来呢？

9.4.5　删除按钮代码设计

单击【删除】按钮后，程序需要完成的操作没有新建和编辑操作那么复杂。它不需要检测记录是否存在，因为该记录一定存在于表中。只有在 ListView 控件中选择了项目后，该按钮才被激活。一旦在 ListView 控件中选择了项目，说明在表中一定有记录存在，因为该项目正是从表中获取的。该按钮的单击事件流程十分简单，代码如下：

```
Private Sub btnDelete_Click()
Dim db As DAO.Database
'链接数据库
Set db = OpenDataBase(ThisWorkbook.Path & "\DB\Info.mdb")
If MultiPage1.SelectedItem.Index Then
      '在产品信息资料表中删除满足条件记录
      db.Execute ("Delete * from t_ProductInfo where Mark='" & ListView.SelectedItem.Text & "' and
Product='" & ListView.SelectedItem.SubItems(1) & "' order by Mark")
      '更新窗口显示
      MultiPage1_Click MultiPage1.SelectedItem.Index
Else
      '在商场名称表中删除满足条件的记录
      db.Execute ("Delete * from t_MarketInfo where MarketName='" & ListView.SelectedItem.Text & "'")
      '更新窗口显示
      MultiPage1_Click MultiPage1.SelectedItem.Index
End If
```

```
'清除临时变量占用空间
Set db = Nothing
End Sub
```

代码说明：

代码中删除记录时，使用了 DAO 数据库对象的 Execute 方法直接执行 SQL 语句。这里执行的正是 Delete 语句。

9.4.6 ListView 控件代码设计

当用户在 ListView 控件中选择了项目时，该选定项目的数据应该立即被显示在多页控件的对应的页面文本框中，以便用户对该记录进行修改操作。该编程操作十分简单。下面是该控件项目单击事件的代码：

```
Private Sub ListView_ItemClick(ByVal Item As MSComctlLib.ListItem)
If MultiPage1.SelectedItem.Index Then
    '单击品牌与产品型号页面时，将 ListView 控件中选定项目的品牌与产品型号写入多页控件的对应
     文本框中
    txtMark.Text = Item.Text                    '写入品牌名称
    txtProduct.Text = Item.SubItems(1)          '写入产品型号
Else
    '单击商场页面时，将 ListView 控件中选定项目的商场名称写入多页控件的文本框中
    txtMarket.Text = Item.Text
End If
'设置各个按钮的状态
btnNew.Enabled = True
btnEdit.Enabled = True
btnDelete.Enabled = True
End Sub
```

9.5 商品销售数据登记窗体设计

商品销售数据登记窗体在系统中是一个多用途窗体。它不仅被用来完成销售数据登记工作，在查询销售记录时，也可以通过该窗体完成编辑操作。该窗体在不同情况下的功能并不冲突，后者使用到的新功能对于前者是无法使用的。

9.5.1 商品销售数据登记窗体界面设计

前面讲过该窗体将被用于两种不同情况，但它们所需要的要素是一样的，因此才有使用同一窗体的事情发生。无论销售数据登记，还是后面查询后的编辑修改操作，都使用到了部分固定数据。这部分数据包括商场名称、品牌名称、产品型号、尺寸、数量、单价 6 个要素。如图 9-25 和图 9-26 所示的是该窗体在不同情况下的界面。



图 9-25　商品销售数据登记窗体界面



图 9-26　编辑记录状态下界面

如图 9-25 所示是在销售数据登记情况下打开的窗体显示情况，其中的【删除】按钮被设置为不可用。如图 9-26 所示的是在编辑记录状态下该窗体的界面，其中的【删除】按钮已经变成可用状态。

表 9-2 列出了该窗体中使用到的各个控件的控件名称、控件类型以及功能说明。

表 9-2　商品销售数据登记窗体控件列表

控 件 名 称	控 件 类 型	功 能 说 明
Frame1	框架控件	该框架控件用于包含资料输入区中所有项目
combMarket	复合框控件	控件显示所有不重复的商场名称列表，用户通过该控件获取对应的商场名称。在其左方有一标签控件提示该控件的作用
combMark	复合框控件	控件显示所有非重复的品牌名称列表，用户通过该控件获取对应的品牌名称
combProduct	复合框控件	控件显示所有非重复的产品型号列表，用户通过该控件获取对应的产品型号
txtSize	文本框控件	控件显示的是对应产品型号的产品尺寸，该文本框是不可编辑的
txtCount	文本框控件	控件用于输入销售商品的数量
txtPrice	文本框控件	控件用于输入销售商品的单价信息
Frame2	框架控件	该控件用于包含窗口的提示信息
btnOK	按钮控件	在销售商品时选择该按钮将将销售商品信息写入商品销售资料表中。当编辑销售信息时，该按钮将完成对商品销售信息的编辑工作
btnDelete	按钮控件	该按钮只在对商品销售信息进行编辑时才能使用，它将把商品的销售资料信息从商品销售资料表中删除
btnCancle	按钮控件	选择该按钮后将退出窗口

建立该窗体的步骤如下：

（1）在 Excel 2007 的 VBE 开发环境中依次选择【插入】|【用户窗体】命令创建一个新窗体。随后在属性窗口中将其名称属性设置为 frmInput。

（2）在工具箱中选择框架控件。在窗体上单击鼠标左键并拖动建立一适当大小的框架控件。然后在属性窗口中将该控件的名称设置为 Frame1。

（3）在工具箱中选择标签控件。然后在 Frame1 框架控件中连续插入 6 个标签控件。随后在属性窗口中设置这些标签的 Caption 属性依次为："商场名称："、"品牌："、"产品型号："、"尺寸："、"数量："和"单价："。

（4）选中所有的标签控件，方法是按住 Ctrl 键依次单击各个标签。然后在属性窗口中将

这些标签的 Width 属性即宽度设置为 50。这个数值的大小可以视具体情况而定。然后在这些标签上右击，在弹出的快捷菜单中依次选择【对齐】|【左对齐】命令，如图 9-27 所示。保持所有控件被选中状态，在 VBE 开发环境中依次选择【格式】|【垂直间距】|【相同】命令，效果如图 9-28 所示。



图 9-27　水平对齐窗体控件

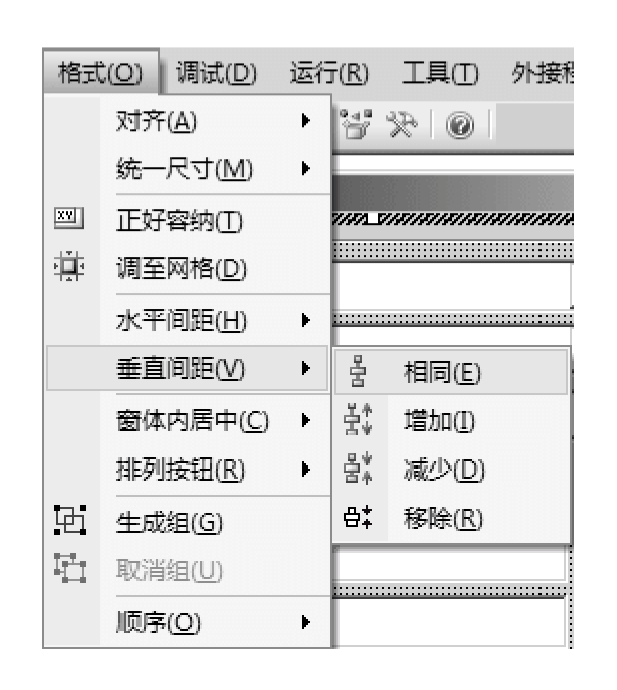


图 9-28　垂直对齐窗体控件

（5）在工具箱中选择复合框控件并在 Frame1 框架控件中连续添加 3 个复合框控件。在属性窗口中设置 3 个复合框控件的名称属性依次为 combMarket、combMark 和 combMark。然后在工具箱中选择文本框控件并添加 3 个文本框控件。在属性窗口中设置 3 个文本框控件的名称属性依次为 txtSize、txtCount 和 txtPrice。

（6）选中所有的复合框与文本框控件，在属性窗口中将这些控件的 SelectionMargin 属性设置为 False。选中的方法与步骤（4）相同，这里不再多做说明。然后右击这些控件，在弹出的快捷菜单中依次选择【对齐】|【左对齐】命令。保持所有控件被选中状态，在 VBE 开发环境中依次选择【格式】|【垂直间距】|【相同】命令。随后再选中所有复合框控件，在属性窗口中设置 Width 属性为 120，该数值视具体情况而定。然后再选中所有文本框控件，在属性窗口中设置 Width 属性为 106。

（7）在工具箱中选择按钮控件并在窗体中连续添加 3 个按钮控件。在属性窗口中设置这些按钮的名称属性依次为 btnOK、btnDelete 和 btnCancel。然后选中并右击刚创建的 3 个按钮，在弹出的快捷菜单中依次选择【统一尺寸】|【两者都相同】命令。再次右击鼠标，在弹出的快捷菜单中依次选择【对齐】|【左对齐】命令。最后在 VBE 开发环境中依次选择【格式】|【垂直间距】|【相同】命令。

（8）在工具箱中选择框架控件并在窗体中插入一框架控件。在属性窗口中将该框架控件的 Caption 属性设置为“提示：”。然后在工具箱中选择标签控件并在刚插入框架控件中创建一标签控件。随后在属性窗口中将该控件的名称属性设置为 labShowMsg，该标签将用于显示提示消息。

9.5.2　窗口初始化、激活与卸载代码设计

本窗体的过程很多，接下来的几个小节没有对各个过程逐一介绍，而是将同类型过程放

置在一起集中介绍。本小节讲述窗口初始化、激活与卸载代码。

在窗口首次被打开时，需要完成初始化工作，这些工作包括设置公共变量以及控件的显示状态及显示内容。窗口中商场名称复合框和品牌名称复合框包含的内容将在窗口被激活事件中完成。将该部分代码放置在窗口被激活事件中，可以保证复合框列表内容在程序运行中能及时反映用户更新后的数据。卸载该窗体时，需要重置部分变量。以下代码分别是窗口初始化、激活与卸载的代码：

```
Private Sub UserForm_Initialize()
'建立到基本信息数据库的链接
Set db = OpenDataBase(ThisWorkbook.Path & "\DB\Info.mdb")
labShowMsg.Caption = "窗口刚被初始化！"                '显示提示信息
btnDelete.Enabled = isEditRecord                        '设置删除按钮的显示状态
'确定当前是输入销售数据还是编辑销售数据
If isEditRecord Then
    Me.Caption = "编辑" & arrEdited(7) & "号记录"       '窗口的标题
    txtCount.Text = arrEdited(4)                        '确定数量文本框数据
    txtPrice.Text = arrEdited(5)                        '确定单价文本框数据
Else
    Me.Caption = "数据输入窗体"
End If
End Sub

Private Sub UserForm_Activate()                         '窗体激活时，设置商场与品牌列表框
SetMarketList                                           '设置商场名称复合框列表内容
ResetMarkList                                           '设置品牌名称复合框列表内容
End Sub

Private Sub UserForm_Terminate()                        '窗体卸载时，清空数据库对象
Set db = Nothing
If isEditRecord Then
    isEditRecord = False
End If
End Sub
```

代码说明：

在初始化过程中，当确认是编辑销售数据时，代码只设置了【数量】文本框和【单价】文本框的数据，其他各个复合框的数据都在窗口被激活事件中被设置。

9.5.3 复合框与文本框改变事件代码设计

用户在使用本窗体时，会根据需要选择适当的商场名称、品牌名称以及产品型号，这些数据都是相互影响的。所有的电器品牌在各个商场都是存在的。一个品牌名称下面有多个产品型号，不同品牌的产品型号不同。用户在品牌复合框中的选择不同，在产品型号复合框中显示的列表内容也是不同的。下面是这些复合框与文本框改变事件的代码：

```
Private Sub combMark_Change()
'当品牌列表框值改变时，如果该值非空，刷新产品型号与尺寸列表框
If Len(combMarket.Text) > 0 Then
    ResetProductList
End If
End Sub

Private Sub combProduct_Change()
当产品型号复合框列表值改变时，刷新产品尺寸文本框的值
txtSize_Change
End Sub

Private Sub txtSize_Change()
Dim strcomProduct As String
strcomProduct = combProduct.Text                          '获取当前产品型号
If Len(strcomProduct) = 0 Then Exit Sub                   '未输入产品型号时退出
'以下代码从产品型号中获取产品尺寸
i = 1
Do Until i = Len(strcomProduct)                           '循环检测产品尺寸字符串各个字符
    j = Asc(Mid(strcomProduct, i, 1))                     '获取第 i 个字符的 ASCII 码
    If j >= 48 And j <= 57 Then                           '检测第 i 个字符是否为数字
        k = Asc(Mid(strcomProduct, i + 1, 1))             '获取第 i+1 个字符的 ASCII 码
        If k >= 48 And k <= 57 Then                       '检测第 i+1 个字符是否为数字
            txtSize.Text = Mid(strcomProduct, i, 2)       '设置尺寸文本框的显示值
            Exit Do
        End If
    End If
    i = i + 1                                             '循环变量加 1
Loop
End Sub
```

代码说明：

- 由于所有商场都有相应的品牌，商场名称复合框的改变并不会引起其他复合框改变。在代码中不用设计商场复合框的改变事件代码。
- 当产品型号复合框发生改变时，需要刷新产品尺寸文本框。程序中将这段代码写入了产品尺寸文本框改变事件中，这样同时也防止用户自己手动输入产品尺寸，无论用户输入什么，程序都根据产品型号自动重新产生产品尺寸。

9.5.4 按钮单击事件代码设计

在该窗体中包含了 3 个功能按钮，分别是【确定】、【删除】和【关闭】按钮。【确定】按钮用于确认输入的商品销售信息或修改信息，【删除】按钮只有在对销售记录进行编辑时可以用，【关闭】按钮用于退出该窗体。下面依次介绍这 3 个按钮的代码。

【确定】按钮较其他两个按钮的代码要复杂，它需要根据具体情况完成相应的操作。如图 9-29 所示的是该按钮单击事件执行流程图。

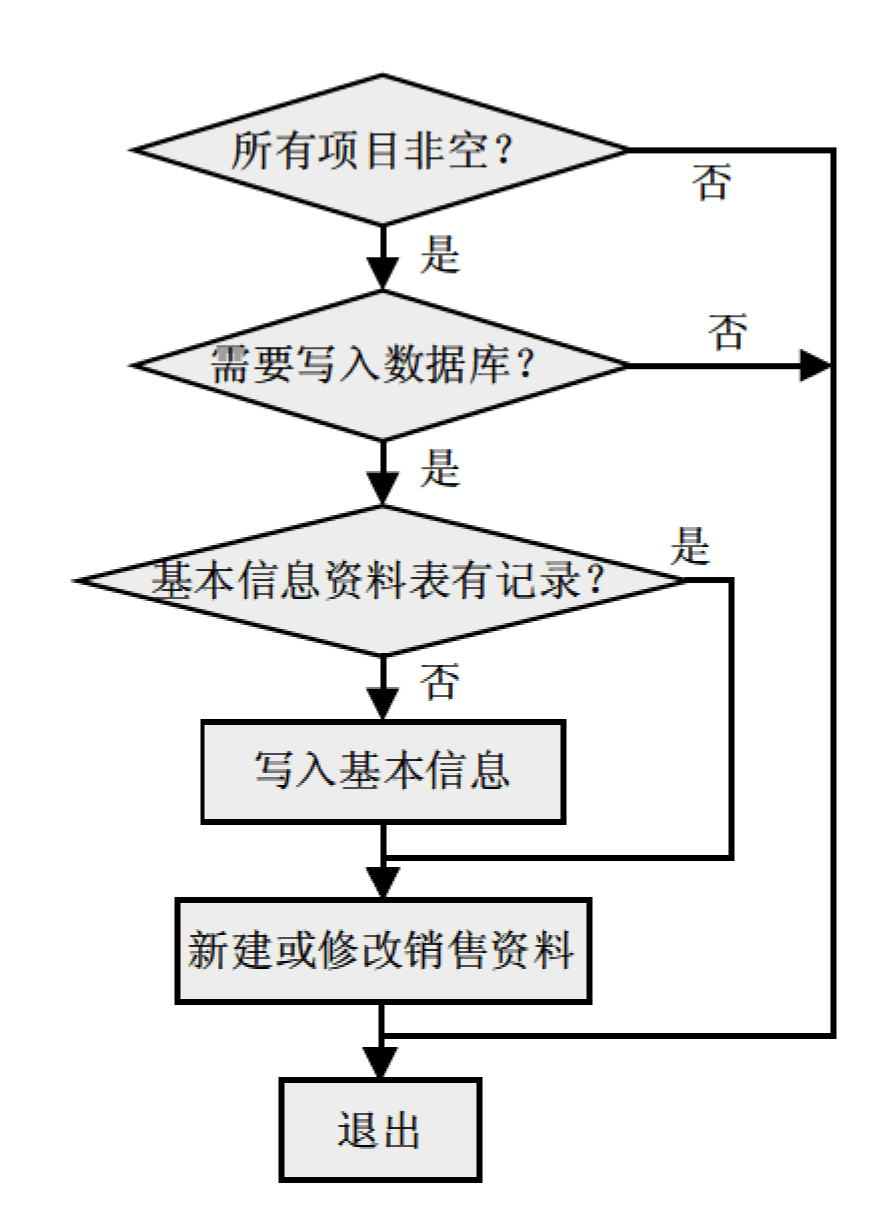


图 9-29　【确定】按钮单击事件程序执行流程图

```
Private Sub btnOK_Click()
Static intClickCount As Integer, rs As DAO.Recordset
Dim isResetMarketList As Boolean, isResetMarkList As Boolean, isResetProductList As Boolean
Dim isToDB As Boolean
'确定窗口输入数据是否写入数据库
isToDB = False
If isEditRecord Then
    '当编辑记录时，将原数据与新数据逐个比较，不一致时表示用户做出了修改，需要写入数据库
    If combMarket.Text <> arrEdited(0) Then        '商场名称是否被修改
        isToDB = True
        GoTo Sub_Next
    End If
    If combMark.Text <> arrEdited(1) Then          '品牌名称是否被修改
        isToDB = True
        GoTo Sub_Next
    End If
    If combProduct.Text <> arrEdited(2) Then       '产品型号是否被修改
        isToDB = True
        GoTo Sub_Next
    End If
    If txtSize.Text <> arrEdited(3) Then           '产品尺寸是否被修改
        isToDB = True
        GoTo Sub_Next
    End If
    If txtCount.Text <> arrEdited(4) Then          '数量是否被修改
        isToDB = True
        GoTo Sub_Next
    End If
    If txtPrice.Text <> arrEdited(5) Then          '单价是否被修改
```

```
            isToDB = True
            GoTo Sub_Next
        End If
Else
        isToDB = True                              '输入销售数据时，直接将数据写入数据库
End If
Sub_Next:
'检查是否有项目为空的情况，所有项目非空时写入数据库
If (Len(Trim(combMarket.Text)) > 0) And (Len(Trim(combMark.Text)) > 0) And
(Len(Trim(combProduct.Text)) > 0) And _
        (Len(Trim(txtSize.Text)) > 0) And (Len(Trim(txtCount.Text)) > 0) And (Len(Trim(txtPrice.Text)) > 0)
Then
        If isToDB Then
            '以下代码分别检测商场名称、品牌以及产品型号在数据库中是否存在
            '检测商品名称在数据库中是否存在
            Set rs = db.OpenRecordset("select * from t_MarketInfo where MarketName='" &
                  combMarket.Text & "'")
            If rs.RecordCount = 0 Then
                i = MsgBox("该买场没有列入数据库，你现在需要将它记录在数据库里吗？", vbOKCancel
                   + vbInformation)
                If i = vbOK Then                   '将新商场名称写入数据库中
                    rs.AddNew
                    rs.Fields("MarketName") = combMarket.Text
                    rs.Update
                    isResetMarketList = True       '标识商场名称复合框需要刷新
                End If
            End If
            '检测品牌名称在数据库中是否存在
            Set rs = db.OpenRecordset("select MarkName from t_MarkInfo where MarkName='" &
                  combMark.Text & "'")
            If rs.RecordCount = 0 Then
                i = MsgBox("该品牌没有列入数据库，你现在需要将它记录在数据库里吗？", vbOKCancel
                   + vbInformation)
                If i = vbOK Then                   '将新品牌名称写入数据库中
                    rs.AddNew
                    rs.Fields("MarkName") = combMark.Text
                    rs.Update
                    isResetMarkList = True         '标识品牌复合框需要刷新
                End If
            End If
            '检测当前品牌下的产品型号在数据库中是否存在
            Set rs = db.OpenRecordset("select * from t_ProductInfo where Mark='" & combMark.Text & "'
                  and Product='" & combProduct.Text & "'")
            If rs.RecordCount = 0 Then
                i = MsgBox("该品牌下对应型号在数据库中没有记录，你现在需要将它记录在数据库里
                   吗？", vbOKCancel + vbInformation)
                If i = vbOK Then
                    With rs                        '将当前品牌下新产品型号写入数据库
                        .AddNew
```

```
                .Fields("Mark") = combMark.Text
                .Fields("Product") = combProduct.Text
                .Fields("Size") = "S" & txtSize.Text
                .Update
            End With
            isResetProductList = True           '标识产品型号复合框需要刷新
        End If
    End If
    '修改商品销售资料表中的数据
    If isEditRecord Then
        RefreshDB 2, combMarket.Text, combMark.Text, combProduct.Text, txtSize.Text,
txtCount.Text, txtPrice.Text
    Else
        RefreshDB 1, combMarket.Text, combMark.Text, combProduct.Text, txtSize.Text,
txtCount.Text, txtPrice.Text
    End If
    intClickCount = intClickCount + 1           '累计单击确定按钮的次数
    labShowMsg.Caption = "记录输入成功，你已经输入了" & intClickCount & "条记录！"
    If isEditRecord Then
        isNeedUpdate = True                     '编辑记录时，标识 ListView 控件项目需要刷新
    End If
  End If
Else
  labShowMsg.Caption = "所输入的信息不能为空！如果你未发现，可能是你输入了空格，请删除空格！"
  MsgBox "请检查输入的信息！信息条目不能为空！", vbOKOnly + vbInformation, "存在空条目"
End If
'刷新各个复合框
If isResetMarketList Then SetMarketList
If isResetMarkList Then ResetMarkList
If isResetProductList Then ResetMarkList
在编辑记录时，单击确定按钮后即刻退出窗口，回到原查询结果窗口中
If isEditRecord Then
  Unload Me
End If
End Sub
```

代码说明：

- 在单击【确定】按钮后，需要检测各个项目是否都输入了数据，当存在空项目时，提示用户输入数据。
- 确认是否需要写入数据库中，检测工作较复杂。在程序中是通过一个布尔变量 isToDB 来标识的，该布尔变量的值在过程最前面获得。对于输入商品销售数据的情况，该变量被直接设置为 True。当对销售数据进行编辑时，程序依次检测各个项目的数据是否发生变化，只要有一项被修改，则将该标识设置为 True。
- 无论是输入销售数据还是编辑销售数据，都要检测当前的基本信息数据是否在基本信息表中建立。程序中对商场名称、品牌名称及产品型号逐个进行检测。当在数据库中没有对应记录时，程序不强制将这些数据写入基本信息资料中，而是通过提示

的方式。

- 新建或修改销售表资料是通过一个公用过程来完成的，即 RefreshDB，该过程在前面已经介绍过。这里对于不同情况，该过程被赋予了不同的参数。

【删除】按钮和【关闭】按钮的代码都十分简单。以下列出两按钮的代码，不再加以详细说明。

```
Private Sub btnDelete_Click()
RefreshDB 3
Unload Me
End Sub

Private Sub btnCancle_Click()
Unload Me
End Sub
```

9.5.5 刷新复合框过程代码设计

在窗口中，商场名称列表、品牌名称列表以及产品型号列表都有可能需要在窗口开启过程中被更新。当用户输入了新的商场名称、品牌名称以及产品型号并确认建立该基本信息后，都需要执行这些过程。这些过程运行方式大致一样，首先从数据库中获取对应的记录集，然后将这些记录依次添加进复合框中。如图 9-30 所示的是这些过程运行的流程图。

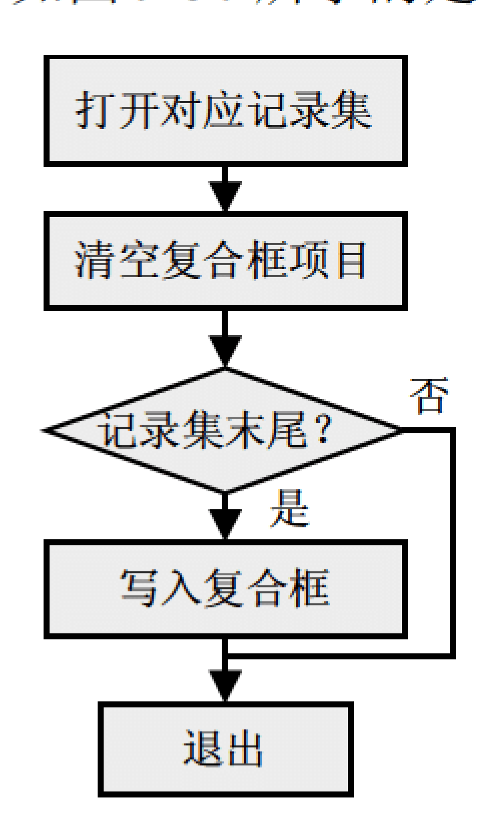


图 9-30　刷新复合框项目流程图

```
Private Sub SetMarketList()                              '刷新商场名称复合框
Dim rs As DAO.Recordset
'从商场名称资料表中获取商场名称记录集
Set rs = db.OpenRecordset("select MarketName from t_MarketInfo order by MarketName",
dbOpenSnapshot, dbReadOnly, dbReadOnly)
With combMarket
    .Clear                                               '清空复合框
    If Not (rs.EOF And rs.BOF) Then
        rs.MoveFirst
        If isEditRecord Then
            .Value = arrEdited(0)                        '编辑完记录，设置复合框的值为编辑值
```

```
            Else
                .Value = rs.Fields("MarketName")        '新建完记录，重置复合框的值为第一个记录值
            End If
            Do Until rs.EOF
                .AddItem rs.Fields("MarketName")        '为复合框添加项目
                rs.MoveNext                             '将记录集指针向下移动
            Loop
        End If
End With
Set rs = Nothing
End Sub

Private Sub ResetMarkList()                             '刷新品牌名称复合框
Dim rs As DAO.Recordset
Set rs = db.OpenRecordset("select MarkName from t_MarkInfo order by MarkName",
dbOpenSnapshot, dbReadOnly, dbReadOnly)
With combMark
        .Clear
        If Not (rs.EOF And rs.BOF) Then
            rs.MoveFirst
            If isEditRecord Then
                .Value = arrEdited(1)
            Else
                .Value = rs.Fields("MarkName")
            End If
            Do Until rs.EOF
                .AddItem rs.Fields("MarkName")
                rs.MoveNext
            Loop
        End If
End With
Set rs = Nothing
End Sub

Private Sub ResetProductList()                          '依据所提供商场列表产品型号列表与尺寸
Dim rs As DAO.Recordset
If Len(combMark.Text) = 0 Then Exit Sub
Set rs = db.OpenRecordset("select Product from t_ProductInfo where Mark='" & combMark.Text & "'
group by Product", dbOpenSnapshot, dbReadOnly, dbReadOnly)
combProduct.Clear
If Not (rs.EOF And rs.BOF) Then
        rs.MoveFirst
        If isEditRecord Then
            combProduct.Value = arrEdited(2)
        Else
            combProduct.Value = rs.Fields("Product")
        End If
        Do Until rs.EOF
            combProduct.AddItem rs.Fields("Product")
```

```
            rs.MoveNext
        Loop
End If
Set rs = Nothing
End Sub
```

代码说明：

- 对于不同的复合框，在过程中打开的记录集是不同的。商场名称记录集可以直接从商场名称资料中获取，品牌名称也可以从品牌名称表中直接获取，但是产品型号不需要从商品资料表中获取对应品牌的产品型号。
- 刷新复合框时，需要设置复合框的值。在程序中，对于输入商品销售数据的情况，复合框的值将被设置为记录集的第一个记录值。对于编辑商品销售数据的情况，复合框的值将被设置为编辑后得到的新记录数据。

9.6　查询销售数据设置窗体设计

查询销售数据设置窗体用于设置销售查询条件。在该窗体中首先需要设置各个单项的查询条件，最终的查询条件是程序自动生成的。在设置了查询条件后，还可以再重新修改查询条件。修改查询条件也是通过修改单项查询条件实现的。

窗体界面设计

在窗体中需要设置的查询条件比较多。相比销售数据输入窗体而言，该查询窗体还添加了一个时间查询条件。通过该日期查询条件，用户可以获得固定销售日期内的销售数据，该窗体的界面如图 9-31 所示。

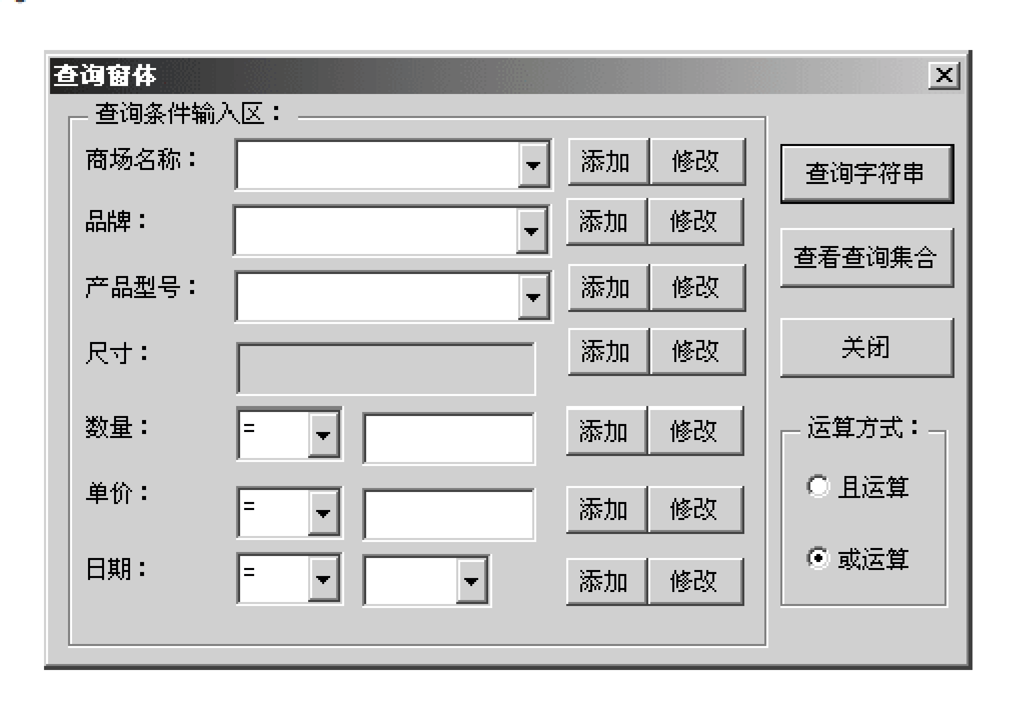


图 9-31　查询销售数据设置窗体界面

对于各个单项，用户可能也需要添加多个查询条件。对于这种情况，在窗体中设置了各个单项查询条件的添加按钮。对于不同条件，只需要单击【添加】按钮即可获得多个条件。表 9-3 列出了在该窗体中使用到的部分控件的名称、控件类型和功能说明。

表 9-3　销售数据查询设置窗体控件列表

控 件 名 称	控 件 类 型	功 能 说 明
combMarket	复合框	该控件列出了所有在数据库中已建立的商场名称信息
combMark	复合框	该控件列出了所有在数据库中已建立的品牌信息
combProduct	复合框	该控件列出了所有在数据库中已建立的产品型号信息
txtSize	文本框	该控件用于设置查询商品尺寸，当产品型号被确定时，该文本框显示的尺寸是对应该产品型号的尺寸。用户不能对该尺寸进行修改
combCompareCount	复合框	该控件列出了所有数量查询运算方式，一共包括了=、<、>、<=、>= 5 种运算方式
txtCount	文本框	该控件设置查询数量的比较数值
combComparePrice	复合框	该控件列出了所有单价查询运算方式，一共包括了=、<、>、<=、>= 5 种运算方式
txtPrice	文本框	该控件设置查询单价的比较数值
combCompareDate	复合框	该控件列出了所有日期查询运算方式，一共包括了=、<、>、<=、>= 5 种运算方式
combDate	复合框	该控件列出了当月所有日期号
btnFilterString	按钮	该按钮打开总查询字符串查看、修改窗口
btnQuery	按钮	该按钮打开查询结果显示窗体
btnClose	按钮	该按钮用于关闭当前窗口
opAND	单选按钮	设置查询条件的逻辑运算方式为与运算
opOR	单选按钮	设置查询条件的逻辑运算方式为或运算
CommandButton+N	按钮	窗口中所有的添加按钮和修改按钮都使用了 CommandButton 加上数字作为名称。这些按钮分别完成各自查询条件的添加和修改工作

在窗体中大量使用了重复控件类型的控件，而且这些相同控件类型的控件被修改的属性也是一致的。下面在介绍窗体建立步骤时，将不对重复控件类型的建立操作做具体介绍。以下是建立该窗体的步骤：

（1）在 Excel 2007 的 VBE 开发环境中依次选择【插入】|【用户窗体】命令。随后在属性窗口中将该新插入窗体的名称属性设置为 frmQuery，如图 9-32 所示。

（2）在工具箱中选择框架控件，在新窗体中单击鼠标左键并拖动以产生适当大小的框架。随后在属性窗口中设置该框架的 Caption 属性为“查询条件输入区：”，如图 9-33 所示。



图 9-32　查询设置窗体属性设置

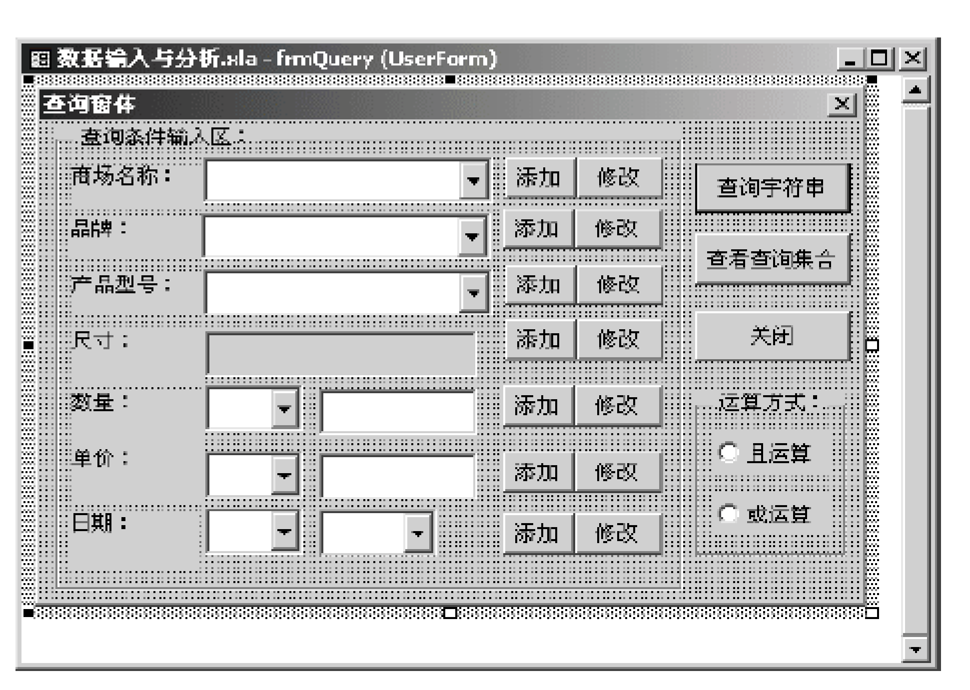


图 9-33　查询窗体设计效果示意图

（3）在工具箱中选择标签控件，然后在框架中连续建立 7 个标签控件。随后在属性窗口中依次设置这些标签控件的 Caption 属性为“商场名称：”、“品牌：”、“产品型号：”、“尺寸：”、“数量：”、“单价：”和“日期：”。

（4）在工具箱中选择复合框控件，在框架中连续建立 7 个复合框控件。随后在属性窗口中依次设置这些复合框的名称属性为 combMarket、combMark、combProduct、combCompareCount、combComparePrice、combCompareDate 和 combDate。SelectionMargin 属性都设置为 False。

（5）在工具箱中选择文本框控件，在框架中连续建立 3 个文本框控件。随后在属性窗口中设置这些文本框的名称属性分别为 txtSize、txtCount 和 txtPrice。SelectionMargin 属性都设置为 False。

（6）在工具箱中选择按钮控件，在框架中连续建立两个按钮控件。在属性窗口中将这两个按钮控件的 Caption 属性依次设置为“添加”和“修改”。然后将这两个按钮控件分别复制 6 份。随后将这些按钮成对放置在各自对应的标签控件右侧。

（7）在工具箱中选择按钮控件，在窗体的右侧连续创建 3 个按钮控件。随后在属性窗口中设置这些按钮的名称属性依次为 btnFilterString、btnQuery 和 btnClose。Caption 属性依次设置为“查询字符串”、“查看查询集合”和“关闭”。

（8）在工具箱中选择框架控件。随后在步骤（7）创建的 3 个按钮控件的下方，新建一个框架控件。然后在属性窗口中将该控件的 Caption 属性设置为“运算方式：”。

（9）在工具箱中选择单选框控件。随后在步骤（8）创建的框架控件中连续创建两个单选按钮。在属性窗口中设置这两个单选按钮控件的名称属性为 opAND 和 opOR，Caption 属性依次设置为“与运算”和“或运算”。

```
Private Sub btnClose_Click()
Unload Me
End Sub

Private Sub btnFilterString_Click()                    '准备信息，然后显示总筛选条件
SetSQL
intFilterIndex = 8
frmFilterEdit.Show
End Sub

Private Sub btnQuery_Click()                           '准备信息，然后显示筛选结果窗体
SetSQL
Me.Hide
frmQryResult.Show
End Sub

Private Sub combMark_Change()                          '当品牌列表变化时，刷新产品型号与尺寸列表
ResetProcutList
End Sub

Private Sub combProduct_Change()
Dim strcomProduct As String
strcomProduct = combProduct.Text
```

```
If Len(strcomProduct) = 0 Then Exit Sub
i = 1
Do Until i = Len(strcomProduct)
    j = Asc(Mid(strcomProduct, i, 1))
    If j >= 48 And j <= 57 Then
        k = Asc(Mid(strcomProduct, i + 1, 1))
        If k >= 48 And k <= 57 Then
            txtSize.Text = Mid(strcomProduct, i, 2)
            Exit Do
        End If
    End If
    i = i + 1
Loop
End Sub

Private Sub txtSize_Change()
Dim strcomProduct As String
strcomProduct = combProduct.Text
If Len(strcomProduct) = 0 Then Exit Sub
i = 1
Do Until i = Len(strcomProduct)
    j = Asc(Mid(strcomProduct, i, 1))
    If j >= 48 And j <= 57 Then
        k = Asc(Mid(strcomProduct, i + 1, 1))
        If k >= 48 And k <= 57 Then
            txtSize.Text = Mid(strcomProduct, i, 2)
            Exit Do
        End If
    End If
    i = i + 1
Loop
End Sub

Private Sub CommandButton1_Click()
intFilterIndex = 2
AddFilter 2
End Sub

Private Sub CommandButton10_Click()
intFilterIndex = 6
AddFilter 6
End Sub

Private Sub CommandButton11_Click()
intFilterIndex = 7
AddFilter 7
End Sub

Private Sub CommandButton12_Click()
```

```
intFilterIndex = 7
frmFilterEdit.Show
End Sub

Private Sub CommandButton2_Click()
intFilterIndex = 2
frmFilterEdit.Show
End Sub

Private Sub CommandButton3_Click()
intFilterIndex = 3
AddFilter 3
End Sub

Private Sub CommandButton4_Click()
intFilterIndex = 3
frmFilterEdit.Show
End Sub

Private Sub CommandButton5_Click()
intFilterIndex = 4
frmFilterEdit.Show
End Sub

Private Sub CommandButton6_Click()
intFilterIndex = 4
AddFilter 4
End Sub

Private Sub CommandButton7_Click()
intFilterIndex = 5
frmFilterEdit.Show
End Sub

Private Sub CommandButton8_Click()
intFilterIndex = 5
AddFilter 5
End Sub

Private Sub CommandButton9_Click()
intFilterIndex = 6
frmFilterEdit.Show
End Sub

Private Sub CommandButton13_Click()          '单击添加单项筛选条件时，设置准备信息
intFilterIndex = 1                           '决定是对哪个单项进行了设置筛选条件操作
AddFilter 1                                  '合并该单项下的筛选条件
End Sub
```

```
Private Sub CommandButton14_Click()                    '显示编辑单项筛选条件窗体，设置准备信息
intFilterIndex = 1
frmFilterEdit.Show
End Sub

Private Sub UserForm_Initialize()                      '初始化窗体
Dim arrCompare As Variant
SetMarketList
ResetMarkList
ResetProcutList
arrCompare = Array("=", "<", ">", "<=", ">=")
combCompareCount.Clear
combComparePrice.Clear
combCompareDate.Clear
For i = 0 To 4
    combCompareCount.AddItem arrCompare(i)
    combComparePrice.AddItem arrCompare(i)
    combCompareDate.AddItem arrCompare(i)
Next
combCompareCount.Text = "="
combComparePrice.Text = "="
combCompareDate.Text = "="
combDate.Clear
For i = 1 To 31
    combDate.AddItem i
Next
opOR.Value = True
End Sub

Private Sub SetMarketList()                            '刷新商场列表
Dim rs As DAO.Recordset
Set db = OpenDataBase(ThisWorkbook.Path & "\DB\Info.mdb")
Set rs = db.OpenRecordset("select MarketName from t_MarketInfo order by MarketName",
dbOpenSnapshot, dbReadOnly, dbReadOnly)
With combMarket
    .Clear
    If Not (rs.EOF And rs.BOF) Then
        rs.MoveFirst
        Do Until rs.EOF
            .AddItem rs.Fields("MarketName")
            rs.MoveNext
        Loop
    End If
End With
Set rs = Nothing
End Sub

Private Sub ResetMarkList()                            '依据所提供商场列表产品型号列表
Dim rs As DAO.Recordset
```

```
Set db = OpenDataBase(ThisWorkbook.Path & "\DB\Info.mdb")
Set rs = db.OpenRecordset("select MarkName from t_MarkInfo order by MarkName",
dbOpenSnapshot, dbReadOnly, dbReadOnly)
With combMark
    .Clear
    If Not (rs.EOF And rs.BOF) Then
        rs.MoveFirst
        Do Until rs.EOF
            .AddItem rs.Fields("MarkName")
            rs.MoveNext
        Loop
    End If
End With
Set rs = Nothing
End Sub

Private Sub ResetProcutList()                    '依据所提供商场列表产品型号列表
Dim rs As DAO.Recordset
If Len(combMark.Text) = 0 Then Exit Sub
Set db = OpenDataBase(ThisWorkbook.Path & "\DB\Info.mdb")
Set rs = db.OpenRecordset("select Product from t_ProductInfo where Mark='" & combMark.Text & "'
Group by Product", dbOpenSnapshot, dbReadOnly, dbReadOnly)
combProduct.Clear
If Not (rs.EOF And rs.BOF) Then
    rs.MoveFirst
    Do Until rs.EOF
        combProduct.AddItem rs.Fields("Product")
        rs.MoveNext
    Loop
End If
Set rs = Nothing
End Sub

Private Sub opAND_Click()                        '设置运算方式为 AND
opMethod = True
End Sub

Private Sub opOR_Click()                         '设置运算方式为 OR
opMethod = False
End Sub

Private Sub AddFilter(intIndex As Integer)       '修改单项筛选条件
Select Case intIndex
    Case 1
        If Len(Trim(combMarket.Text)) > 0 Then
            If Len(strMarketFilter) Then
                If opMethod Then
                    strMarketFilter = strMarketFilter & " and MarketName='" &
Trim(combMarket.Text) & "'"
```

```
                Else
                    strMarketFilter = strMarketFilter & " or MarketName='" & Trim(combMarket.Text) & "'"
                End If
            Else
                strMarketFilter = "MarketName='" & Trim(combMarket.Text) & "'"
            End If
        End If
    Case 2
        If Len(Trim(combProduct.Text)) Then
            If Len(strProductFilter) Then
                If opMethod Then
                    strProductFilter = strProductFilter & " and Product='" & Trim(combProduct.Text) & "'"
                Else
                    strProductFilter = strProductFilter & " or Product='" & Trim(combProduct.Text) & "'"
                End If
            Else
                strProductFilter = "Product='" & Trim(combProduct.Text) & "'"
            End If
        End If
    Case 3
        If Len(Trim(txtSize.Text)) Then
            If Len(strSizeFilter) Then
                If opMethod Then
                    strSizeFilter = strSizeFilter & " and Size='" & Trim(txtSize.Text) & "'"
                Else
                    strSizeFilter = strSizeFilter & " and Size='" & Trim(txtSize.Text) & "'"
                End If
            Else
                strSizeFilter = "Size='" & Trim(txtSize.Text) & "'"
            End If
        End If
    Case 4
        If Len(Trim(txtCount.Text)) > 0 Then
            If Len(strNumberFilter) Then
                If opMethod Then
                    strNumberFilter = strNumberFilter & " and Number" & combCompareCount.Text
                        & Trim(txtCount.Text)
                Else
                    strNumberFilter = strNumberFilter & " or Number" & combCompareCount.Text &
                        Trim(txtCount.Text)
                End If
            Else
                strNumberFilter = "Number" & combCompareCount.Text & Trim(txtCount.Text)
            End If
        End If
    Case 5
        If Len(Trim(txtPrice.Text)) Then
            If Len(strPriceFilter) Then
                If opMethod Then
                    strPriceFilter = strPriceFilter & " and Price" & combComparePrice.Text &
```

```
Trim(txtPrice.Text)
                Else
                    strPriceFilter = strPriceFilter & " or Price" & combComparePrice.Text &
                        Trim(txtPrice.Text)
                End If
            Else
                strPriceFilter = "Price" & combComparePrice.Text & Trim(txtPrice.Text)
            End If
        End If
    Case 6
        If Len(Trim(combDate.Text)) Then
            If Len(strDateFilter) Then
                If opMethod Then
                    strDateFilter = strDateFilter & " and Date" & combCompareDate.Text &
                        Trim(combDate.Text)
                Else
                    strDateFilter = strDateFilter & " or Date" & combCompareDate.Text &
                        Trim(combDate.Text)
                End If
            Else
                strDateFilter = "Date" & combCompareDate.Text & Trim(combDate.Text)
            End If
        End If
    Case 7
        If Len(Trim(combMark.Text)) Then
            If Len(strMarkFilter) Then
                If opMethod Then
                    strMarkFilter = strMarkFilter & " and Mark='" & Trim(combMark.Text) & "'"
                Else
                    strMarkFilter = strMarkFilter & " or Mark='" & Trim(combMark.Text) & "'"
                End If
            Else
                strMarkFilter = "Mark='" & Trim(combMark.Text) & "'"
            End If
        End If
End Select
End Sub

Private Sub UserForm_Terminate()                '窗体卸载时，清空各个变量
strMarketFilter = ""
strProductFilter = ""
strSizeFilter = ""
strNumberFilter = ""
strPriceFilter = ""
strDateFilter = ""
strMarkFilter = ""
Set db = Nothing
End Sub
```

9.7 查询显示窗体设计

查询显示与分析窗体用于显示满足查询设置窗体中设置条件的记录集。窗体通过一个 ListView 控件显示了所有满足条件的记录。在该窗体中用户可以继续对销售记录进行编辑，还可以将这些记录导出到新的 Excel 表中，也可以在该查询结果中再次手动进行筛选并且导出筛选后的结果。

9.7.1 窗体界面设计

查询结果显示窗体包含了一个 ListView 控件用于显示查询结果。窗体还包含了 5 个功能按钮，这 5 个功能按钮分别是导出所有项、重置、仅显示勾选项、编辑和关闭。如图 9-34 所示的是该窗体的界面。

- 导出所有项：该按钮将把 ListView 控件中显示的所有记录项目导出到一个新 Excel 工作簿中。当用户通过手动选择项目后，单击【仅显示勾选项】按钮后，【导出所有项】按钮导出的数据是选中后显示出来的所有数据。
- 重置：该按钮只有在单击【仅显示勾选项】按钮后才被激活。该按钮被单击后，ListView 控件的显示记录将被恢复到初始查询结果。
- 仅显示勾选项：该按钮把 ListView 中被选中的项目单独列出显示在 ListView 控件中。
- 编辑：该按钮将打开记录编辑窗口，它相当于在项目上双击的效果。
- 关闭：该按钮将关闭查询显示结果窗体。



图 9-34 查询显示窗体界面

窗体的界面布局十分简单，需要注意的是当 ListView 控件在工具箱中无法找到时，需要引用 Microsfot Windows Common Controls 6.0(SP6)。在前面章节已经接触到该方面的内容，具体操作方法请见相应章节的介绍。

9.7.2 窗体事件代码设计

在该窗体中，有关窗体的事件包括窗体的初始化以及窗体卸载事件。窗口初始化事件除了将查询到的记录集显示到 ListView 控件上，还需要设置各个按钮的可用状态。窗体卸载时，需要删除临时表、清除部分变量占用内存空间以及恢复查询设置窗体显示。初始化的代码比较多，下面着重讲述该部分代码。如图 9-35 所示的是该初始化事件过程的流程图。

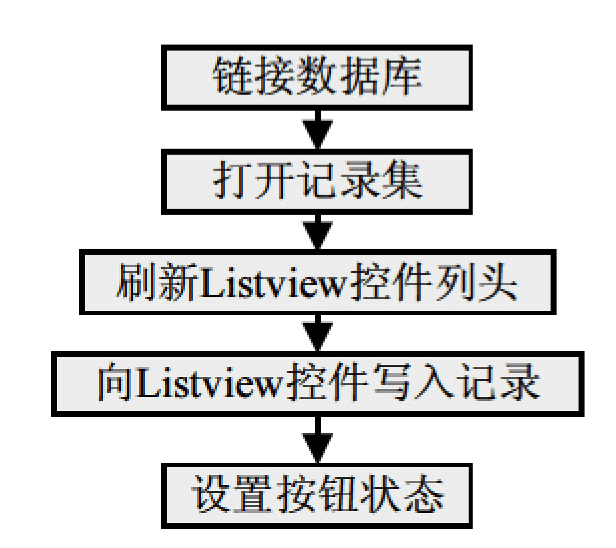


图 9-35　查询结果显示窗体初始化事件代码执行流程图

```
Private Sub UserForm_Initialize()                '窗体初始化时，建立数据库对象与记录集对象
Dim qryString As String
Dim itemList As ListItem, rsCount As Long
'链接数据库
Set db = OpenDataBase(ThisWorkbook.Path & "\DB\db.mdb")
'打开记录集
If Len(strSQL) > 0 Then                          '确定查询记录集的 SQL 语句
    qryString = "select * from t_Sale where " & strSQL
Else
    qryString = "select * from t_Sale"
End If
'以快照、只读形式打开记录集
Set rs = db.OpenRecordset(qryString, dbOpenSnapshot, dbReadOnly, dbReadOnly)
isSaveAll = True
'刷新 ListView 控件列头
refreshListHeader
'将记录集写入 ListView 控件中
If Not (rs.EOF And rs.BOF) Then
    rs.MoveFirst
    ListViewQry.LabelEdit = lvwManual            '单击 ListView 控件第一列时不变成被编辑状态
    ListViewQry.HideSelection = False            'ListView 失去焦点时仍然显示被选择项
    With ListViewQry.ListItems
        .Clear
        Do Until rs.EOF
            Set itemList = .Add(Text:=rs.Fields("MarketName"))
            itemList.SubItems(1) = rs.Fields("Mark")
            itemList.SubItems(2) = rs.Fields("Product")
            itemList.SubItems(3) = rs.Fields("Size")
            itemList.SubItems(4) = rs.Fields("Number")
            itemList.SubItems(5) = rs.Fields("Price")
            itemList.SubItems(6) = rs.Fields("Date")
            itemList.SubItems(7) = rs.Fields("ID")
            rsCount = rsCount + 1
            rs.MoveNext
        Loop
    End With
```

```
End If
'ListView 控件项目将按照第一列升序排列
With ListViewQry
        .SortKey = 0
        .Sorted = True
        .SortOrder = lvwAscending
End With
'在窗体的标题栏显示提示信息
Me.Caption = "查询结果 -总共" & rsCount & "条记录"
'设置按钮可用状态
btnReset.Enabled = False
End Sub

Private Sub UserForm_Terminate()            '窗体卸载时，清除数据库，记录集对象，删除临时文件
On Error Resume Next
db.Execute ("Drop Table TempTable")
On Error GoTo 0
Set rs = Nothing
Set db = Nothing
frmQuery.Show
End Sub
```

代码说明：

- 程序中，当没有设置查询条件时，默认的是查询所有记录。在初始化过程打开记录集时，首先使用了 If 语句分别确定了对应的 SQL 查询语句，以便得出满足要求的记录集。
- 程序中的记录集是以快照、只读形式打开的，这种形式的记录集不可被修改。程序中只需要读取记录集数据，使用该种形式打开记录集可以加快运行速度。
- 当希望 ListView 控件显示结果按照某列进行排序时，首先需要指定排序的列索引 SortKey，该值从 0 开始计数，然后将 Sorted 设置为 True，开启排序。排序的顺序由 SortOrder 决定。
- 在初始化窗体过程中，调用了 refreshListHeader 过程，该过程用于刷新 ListView 控件的列头。该过程的代码如下：

```
Private Sub refreshListHeader()                   '刷新列表头
With ListViewQry
        .Gridlines = True                         '显示网格线
        .FullRowSelect = True                     '允许选中整行
        .MultiSelect = True                       '允许多行选择
        .LabelEdit = False                        '不允许编辑标签控件
        .View = lvwReport                         '预览模式为 lvwReport
        With .ColumnHeaders
                .Clear                            '清除列头
                .Add Text:="商场名", Width:=74    '以下分别添加各列列头
                .Add Text:="品牌", Width:=39
                .Add Text:="产品型号", Width:=65
                .Add Text:="尺寸", Width:=35
```

```
            .Add Text:="数量", Width:=35
            .Add Text:="单价", Width:=49
            .Add Text:="日期", Width:=30
            .Add Text:="ID", Width:=74
        End With
End With
End Sub
```

9.7.3 ListView 控件事件代码设计

窗体的主体部分就是 ListView 控件，窗体中的按钮执行的命令也是与该 ListView 控件相关联的。本部分讲述的代码是该控件的事件代码，包括列单击事件、控件双击事件、项目单击事件。这些事件都是为了响应用户的操作而建立的。

列单击事件可以支持自定义排序。初始化时 ListView 控件按照第一列进行升序排列，但用户可能需要将记录按照其他的列进行排序以便于快速定位到需要查看的记录。ListView 本身不支持该功能，但通过列单击事件可以实现相应的功能。

ListView 控件支持双击打开某项记录的编辑窗口，但是 ListView 控件并没有项目的双击事件。程序实现该功能的方式是通过 ListView 控件的双击事件完成的。当 ListView 控件被双击时，将打开在控件中被选择项目的编辑窗口，否则任何动作都不执行。以下是这些事件的详细代码：

```
Private Sub ListViewQry_ColumnClick(ByVal ColumnHeader As MSComctlLib.ColumnHeader)
                                                    '单击列表头时排序
Static OldIndex As Integer, CountNum As Integer
OldIndex = ListViewQry.SortKey                      '记录原排序列的索引号
ListViewQry.SortKey = ColumnHeader.Index - 1        '获取当前需排序列的列索引号
ListViewQry.Sorted = True                           '开启排序
If OldIndex = ListViewQry.SortKey Then              'CountNum 记录在同一列单击的次数
    CountNum = CountNum + 1
Else
    CountNum = 0
End If
If CountNum Mod 2 = 0 Then                          '按照单击同一列的次数确定排序的方式
    ListViewQry.SortOrder = lvwAscending
Else
    ListViewQry.SortOrder = lvwDescending
End If
End Sub

Private Sub ListViewQry_DblClick()
If Not ListViewQry.SelectedItem Is Nothing Then
    Set itemEdited = ListViewQry.SelectedItem       '将当前选择的项目赋给对象变量
    btnEdit_Click                                   '开启销售记录编辑窗体
End If
End Sub
```

9.7.4　导出所有项按钮代码设计

【导出所有项】按钮将把 ListView 控件中显示的所有记录保存到 Excel 文件中。如图 9-36 所示的是该过程运行的流程图。

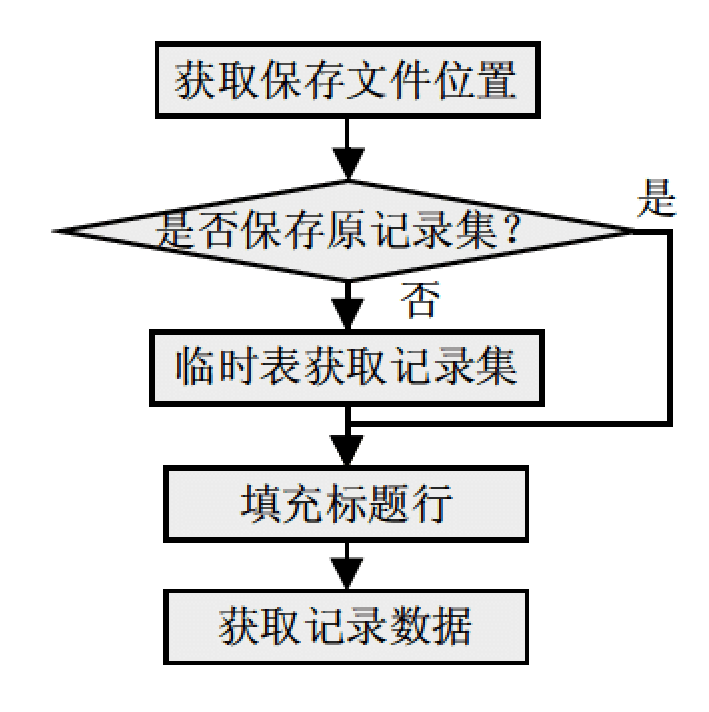


图 9-36　【导出所有项】按钮运行流程图

```
Private Sub btnToXLS_Click()                    '导出所有记录到 Excel 文件
Dim wk As Workbook, ws As Worksheet, strFileSaveName As String, rsXLS As DAO.Recordset
'创建 CommonDialog 对象
Set objDialog = CreateObject("UserAccounts.CommonDialog")
objDialog.Filter = "Excel File(*.xls)|*.xls"    '设置文件类型
objDialog.FilterIndex = 1                       '首次打开文件筛选器时，默认的文件类型为第一个
intResult = objDialog.ShowOpen                  '打开文件筛选器
strFileSaveName = objDialog.FileName            '将文件名称保存在变量中
If Len(strFileSaveName) > 0 Then
    Set wk = Workbooks.Add                      '建立新工作簿以保存数据
    Application.DisplayAlerts = False
    Do Until wk.Worksheets.Count = 1            '删除无用工作表，只留下一个工作表且命名为“查询结果”
        wk.Worksheets(1).Delete
    Loop
    Application.DisplayAlerts = True
    Set ws = wk.Worksheets(1)
    ws.Name = "查询结果"
    '当保存筛选后的数据时，从临时表获取记录集。否则使用原查询记录集
    If Not isSaveAll Then
        Set db = OpenDataBase(ThisWorkbook.Path & "\DB\db.mdb")
        Set rsXLS = db.OpenRecordset("TempTable")
    End If
    rs.MoveFirst
    With ws
        '为数据表添加列头
        .Cells(1, 1) = "商场名"
        .Cells(1, 2) = "品牌"
        .Cells(1, 3) = "产品型号"
        .Cells(1, 4) = "尺寸"
        .Cells(1, 5) = "数量"
```

```
            .Cells(1, 6) = "单价"
            .Cells(1, 7) = "日期"
            .Cells(1, 8) = "ID"
            If Not isSaveAll Then
                .Range("A2").CopyFromRecordset rsXLS     '当保存筛选后的记录时，复制临时表获取记
                                                          录集的数据
            Else
                .Range("A2").CopyFromRecordset rs         '当保存查询结果记录时，复制原查询记录集
                                                          的数据
            End If
            .Columns("A:H").EntireColumn.AutoFit          '将新表的列自动对齐
        End With
        wk.SaveAs strFileSaveName                         '保存文件
        wk.Application.Visible = True
        MsgBox "文件保存成功！ ", vbOKOnly + vbInformation, "保存"
End If
Set ws = Nothing
Set wk = Nothing
Set rsXLS = Nothing
End Sub
```

9.7.5 重置按钮代码设计

当用户在 ListView 控件中选择了部分记录项后，单击【仅显示勾选项】按钮后，【重置】按钮被激活。该按钮用于将 ListView 控件显示项目恢复到原来的查询结果记录状态。代码十分简单，它直接调用了窗体的初始化事件过程。因为该过程完成的就是显示查询结果记录并且修改按钮显示状态。在该按钮被单击后，IsSaveAll 被设置为 True，因为此时显示的是所有查询记录。

```
Private Sub btnReset_Click()
UserForm_Initialize
isSaveAll = True
End Sub
```

9.7.6 仅显示勾选项按钮代码设计

该按钮将把用户在 ListView 控件中选择的记录单独显示在 ListView 控件中。该按钮的单击事件首先检测是否有记录被选中。当有记录被选中时，程序将把所有已选中的项目显示在 ListView 控件中，并且将这些记录数据保存在临时表中。这部分的工作是由过程 ShowSelectedItem 完成的。下面是该按钮的单击事件代码，过程 ShowSelectedItem 的详细代码将在后面详细介绍。

```
Private Sub btnShowSelected_Click()              '显示被选中的项目
Dim isSelectedExist As Boolean
'检查是否有项目被选中
```

```
isSelectedExist = False
For Each clhListItem In ListViewQry.ListItems
    If clhListItem.Checked = True Then     '当发现有项目被选中时，标记 isSelectedExist，并退出
                                            检测循环
        isSelectedExist = True
        Exit For
    End If
Next
'显示选中记录
If isSelectedExist Then
    ShowSelectedItem                       '在 ListView 控件中显示被选中的所有记录
    btnReset.Enabled = True                '设置重置按钮的可用状态
    isSaveAll = False                      '此时保存数据是对手动选中后的数据，而非原查询数据
Else
    MsgBox "没有勾选任何选项！ ", vbOKOnly + vbInformation, "提示"
End If
End Sub
```

将所有已选中的记录项目显示在 ListView 控件中并将数据保存在临时表的 ShowSelectedItem 过程。其代码比较复杂，这里加以详细介绍。图 9-37 所示的是该过程的执行流程图。

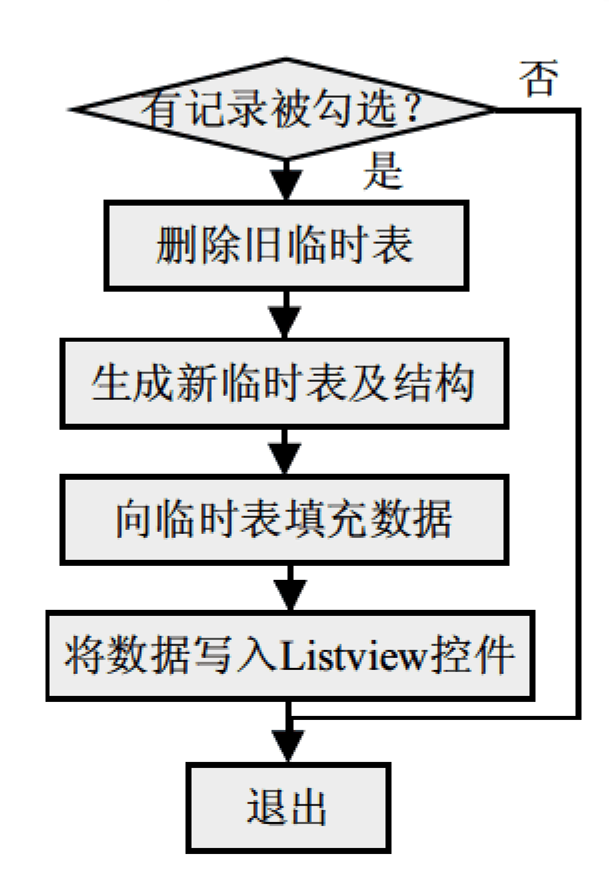


图 9-37　仅显示勾选项过程流程图

```
Private Sub ShowSelectedItem()     '在列表中显示选中项目（保存选中项目到临时文件，然后刷新列表）
Dim clhListItem As ListItem, isSelectedExist As Boolean
Dim rsXLS As DAO.Recordset, tdf As DAO.TableDef
isSelectedExist = False
'检测是否有记录被选中
For Each clhListItem In ListViewQry.ListItems
    If clhListItem.Checked = True Then
        isSelectedExist = True
        Exit For
    End If
Next
'删除旧临时表并生成新临时表
If isSelectedExist Then
    On Error Resume Next
```

```
db.Execute ("Drop Table TempTable")
On Error GoTo 0
Set db = OpenDataBase(ThisWorkbook.Path & "\DB\db.mdb")          '链接数据库
Set tdf = db.CreateTableDef("TempTable")                          '创建临时表及其字段
Set fld = tdf.CreateField("MarketName", dbText, 50)
tdf.Fields.Append fld
Set fld = tdf.CreateField("Mark", dbText, 50)
tdf.Fields.Append fld
Set fld = tdf.CreateField("Product", dbText, 50)
tdf.Fields.Append fld
Set fld = tdf.CreateField("Size", dbText, 50)
tdf.Fields.Append fld
Set fld = tdf.CreateField("Number", dbInteger)
tdf.Fields.Append fld
Set fld = tdf.CreateField("Price", dbDouble)
tdf.Fields.Append fld
Set fld = tdf.CreateField("Date", dbInteger)
tdf.Fields.Append fld
Set fld = tdf.CreateField("ID", dbText, 14)
tdf.Fields.Append fld
db.TableDefs.Append tdf
Set rsXLS = db.OpenRecordset("TempTable")                         '向临时表添加记录
For Each clhListItem In ListViewQry.ListItems
    If clhListItem.Checked = True Then
        rsXLS.AddNew
        rsXLS.Fields("MarketName") = clhListItem.Text
        rsXLS.Fields("Mark") = clhListItem.SubItems(1)
        rsXLS.Fields("Product") = clhListItem.SubItems(2)
        rsXLS.Fields("Size") = clhListItem.SubItems(3)
        rsXLS.Fields("Number") = clhListItem.SubItems(4)
        rsXLS.Fields("Price") = clhListItem.SubItems(5)
        rsXLS.Fields("Date") = clhListItem.SubItems(6)
        rsXLS.Fields("ID") = clhListItem.SubItems(7)
        rsXLS.Update
        i = i + 1
    End If
Next
'清空 ListView 控件后将勾选记录显示在 ListView 控件上
rsXLS.MoveFirst
With ListViewQry.ListItems
    .Clear
    Do Until rsXLS.EOF
        Set clhListItem = .Add(Text:=rsXLS.Fields("MarketName"))
        clhListItem.SubItems(1) = rsXLS.Fields("Mark")
        clhListItem.SubItems(2) = rsXLS.Fields("Product")
        clhListItem.SubItems(3) = rsXLS.Fields("Size")
        clhListItem.SubItems(4) = rsXLS.Fields("Number")
        clhListItem.SubItems(5) = rsXLS.Fields("Price")
        clhListItem.SubItems(6) = rsXLS.Fields("Date")
        clhListItem.SubItems(7) = rsXLS.Fields("ID")
```

```
                rsXLS.MoveNext
            Loop
        End With
End If
Me.Caption = "勾选后的结果 -总共" & i & "条记录"
Set rsXLS = Nothing
Set dbXLS = Nothing
Set wkJet = Nothing
End Sub
```

9.7.7　编辑按钮代码设计

单击【编辑】按钮时，首先需要获取记录的相关信息。程序通过一个 arrEdited 数组保存了选定记录项的所有数据。当打开记录编辑窗口后，该窗口将由此确定需要显示并编辑的记录。关于编辑窗体如何利用这个数组完成编辑操作的代码请见 9.5 节的介绍。以下是该按钮的单击事件代码：

```
Private Sub btnEdit_Click()
If itemEdited Is Nothing Then
    MsgBox "你没有选中任何记录！ ", vbOKOnly + vbInformation
Else
    '将选择项的数据依次保存在全局数组 arrEdited 中
    With frmQryResult.ListViewQry.SelectedItem
        arrEdited =
Array(.Text, .SubItems(1), .SubItems(2), .SubItems(3), .SubItems(4), .SubItems(5), .SubItems(6), .SubItems(7))
    End With
    isEditRecord = True                    '设置当前处于编辑销售记录状态
    frmInput.Show                          '开启销售记录编辑窗口
End If
End Sub
```

9.7.8　关闭按钮代码设计

关闭窗口的代码十分简单，只需使用 Unload 卸载窗口即可。下面是该按钮的代码：

```
Private Sub btnClose_Click()
Unload Me
End Sub
```

9.8　编辑查询条件窗口设计

编辑查询条件窗口用于查看或编辑单项查询条件或总查询条件。当在查询销售数据设置窗体中单击相应查询条件的【编辑】按钮时，将打开相应查询条件的编辑窗口。

9.8.1 窗体界面设计

窗体的界面十分简洁，包含了 1 个文本框控件、2 个按钮。文本框控件用于显示被编辑的查询条件。两个按钮分别用于执行编辑工作和退出窗口。如图 9-38 所示的是该窗口的界面。

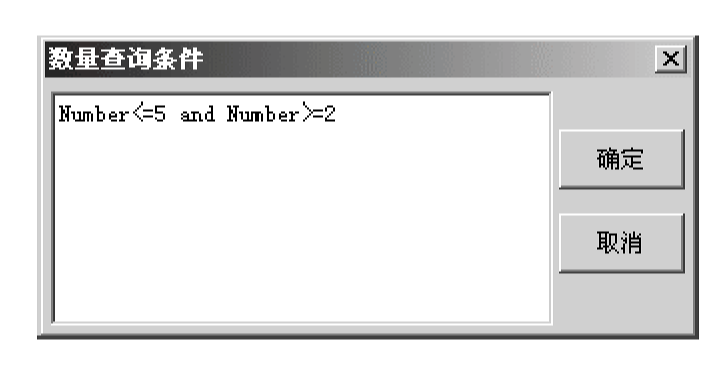


图 9-38　编辑查询条件窗口界面

窗口的界面十分简单，这里不再对包含的控件一一说明。界面的设计过程这里也不再说明，读者可以参照前面的窗口界面设计过程制作。需要说明的是各个控件的命名，文本框控件名称为 txtFilterArea，【确定】按钮名称为 btnOK，【取消】按钮名称为 btnClose。

9.8.2 窗体事件代码设计

窗体的事件包括了窗体激活事件和窗体卸载事件。窗体激活事件将根据被选择项目决定窗口中显示的查询条件是哪个项目的。窗体卸载时需要重置 SQL 查询语句以防干扰其他查询条件的编辑，另外还需要激活原查询设置窗体。两个事件的执行流程都十分简单，这里不列出过程的流程图。

以下是这两个事件的代码：

```
Private Sub UserForm_Activate()                    '窗体被激活时，设置文本框显示内容
With frmFilterEdit
    Select Case intFilterIndex                     'intFilterIndex 在单击编辑按钮时已被确定
        Case 1
            Me.Caption = "商场查询条件"
            .txtFilterArea.Text = strMarketFilter  'strMarketFilter 在查询设置窗口中被确定
        Case 2
            Me.Caption = "产品型号查询条件"
            .txtFilterArea.Text = strProductFilter 'strProductFilter 在查询设置窗口中被确定
        Case 3
            Me.Caption = "尺寸查询条件"
            .txtFilterArea.Text = strSizeFilter    'strSizeFilter 在查询设置窗口中被确定
        Case 4
            Me.Caption = "数量查询条件"
            .txtFilterArea.Text = strNumberFilter  'strNumberFilter 在查询设置窗口中被确定
        Case 5
            Me.Caption = "价格查询条件"
            .txtFilterArea.Text = strPriceFilter   'strPriceFilter 在查询设置窗口中被确定
        Case 6
            Me.Caption = "时间查询条件"
            .txtFilterArea.Text = strDateFilter    'strDateFilter 在查询设置窗口中被确定
        Case 7
            Me.Caption = "品牌查询条件"
            .txtFilterArea.Text = strMarkFilter    'strMarkFilter 在查询设置窗口中被确定
```

```
            Case 8
                Me.Caption = "总体查询条件"
                Me.btnOK.Enabled = False                '总查询条件不能被直接编辑，只能修改各子项目
                .txtFilterArea.Text = strSQL
        End Select
End With
End Sub

Private Sub UserForm_Terminate()
strSQL = ""                                             '重置 SQL 查询语句
frmQuery.Show                                           '显示查询设置窗体
End Sub
```

代码说明：

- 窗口初始化事件中使用到的公共变量在查询条件设置窗口中被设置。这些变量包括 intFilterIndex、strMarketFilter、strProductFilter、strSizeFilter、strNumberFilter、strPriceFilter、strDateFilter 和 strMarkFilter。
- 为了确定窗口最终显示的哪一个分项的查询条件，程序使用了一个 Select Case 语句。针对 intFilterIndex 参数的值决定窗口显示内容以及按钮的状态。

9.8.3　文本框改变事件

当窗口打开的是总查询字符串时，窗口的【确定】按钮是灰色的，即总查询字符串是不能在编辑窗口中修改的。当需要修改时，需要修改相应子项目的查询条件。但用户可能会在文本框中试图改变总查询字符串。文本框改变事件正是用于检测用户这一动作，以显示提示信息，提示用户在子项目中修改条件。该文本框改变事件的代码如下：

```
Private Sub txtFilterArea_Change()
If txtFilterArea.Text <> strSQL And intFilterIndex = 8 Then        '如果是针对 strSQL 进行修改时，提示
                                                                    非法操作
    txtFilterArea.Text = strSQL
    MsgBox "总体查询条件不能在此修改，你只能在单项目的修改按钮种做该动作！", vbOKOnly +
vbInformation
End If
End Sub
```

9.8.4　确定按钮代码设计

单击【确定】按钮后，程序将根据 intFilterIndex 全局变量确定被修改的单项查询条件的归属，然后对应查询条件修改为文本框中的值。总查询字符串是根据单项字符串获取的，因此这里不需要考虑总查询字符串的修改情况。实际上总查询字符串在编辑窗口中也是无法被修改的。

```
Private Sub btnOK_Click()                       '编辑单项筛选条件
With frmFilterEdit
```

```
        Select Case intFilterIndex                    '由 intFilterIndex 区分是针对哪个单项筛选条件
            Case 1
                strMarketFilter = .txtFilterArea.Text
            Case 2
                strProductFilter = .txtFilterArea.Text
            Case 3
                strSizeFilter = .txtFilterArea.Text
            Case 4
                strNumberFilter = .txtFilterArea.Text
            Case 5
                strPriceFilter = .txtFilterArea.Text
            Case 6
                strDateFilter = .txtFilterArea.Text
            Case 7
                strMarkFilter = .txtFilterArea.Text
        End Select
    End With
    Unload Me
    End Sub
```

9.8.5 关闭按钮代码设计

【关闭】按钮用于卸载窗体，代码十分简单，使用 Unload 卸载掉窗口即可。代码如下：

```
Private Sub btnClose_Click()
Unload Me
End Sub
```

9.9 系 统 测 试

本节是系统的测试部分，该部分测试内容包含销售数据的输入、查询和编辑 3 个部分。这里假设基本数据已经建立，系统中也已经建立了部分的基础数据信息。以下将通过 3 个小节分别介绍 3 部分内容的测试过程。

9.9.1 销售数据输入

销售数据的输入十分简便，没有过程操作。在 Excel 2007 的加载宏菜单中依次选择【数据输入与查询】|【输入数据窗口】命令，在随后打开的【数据输入窗体】对话框中依次输入商品名称、品牌、产品型号、数量和单价信息，最后单击【确定】按钮即可，如图 9-39 所示。



图 9-39 【数据输入窗体】对话框

9.9.2　查询销售数据

（1）在 Excel 2007 的加载宏菜单中依次选择【数据输入与查询】|【数据库查询】命令，随后系统打开【查询窗体】对话框，如图 9-40 所示。此处首先查询数据库所有记录数据，因而不需要设置任何查询条件，直接单击【查看查询集合】按钮即可，如图 9-41 所示。

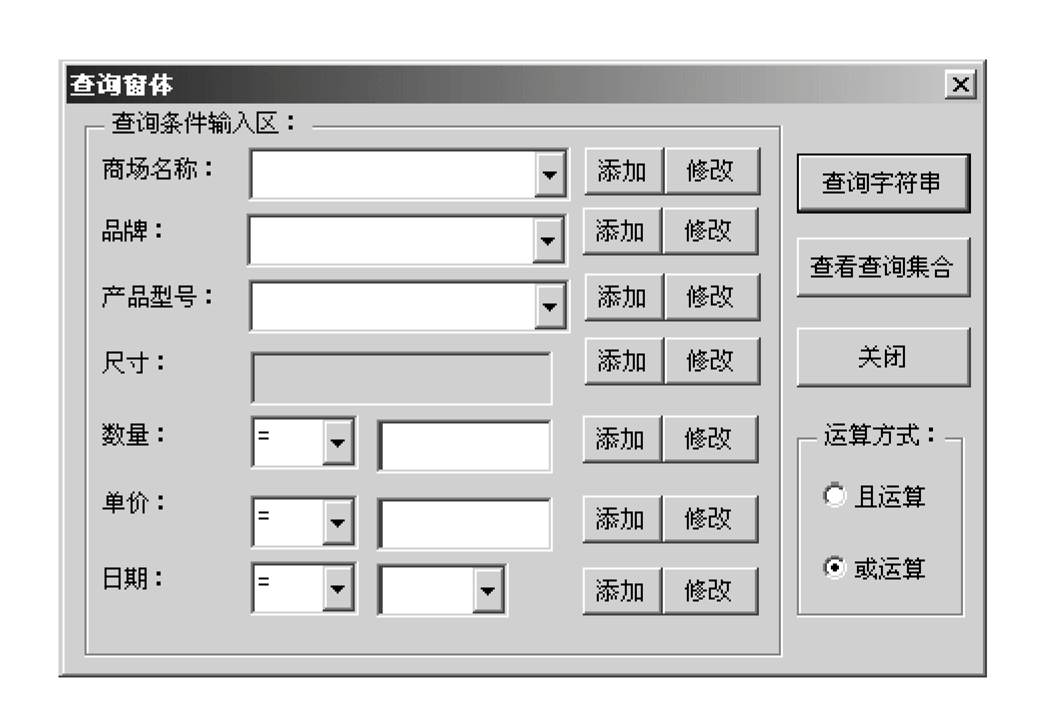


图 9-40　【查询窗体】对话框



查询结果 － 总共8条记录

商场名	品牌	产品型号	尺寸	数量	单价	日期	ID
大润发春申	LG	32LB1R	32	2	1000	8	20070508122027
大润发春申	LG	32LB1R	32	3	200	8	20070508122033
大润发春申	LG	32LB1R	32	10	20000	8	20070508122038
大润发春申	创维	21N15AA	21	2	33	29	20070529193611
大润发闸北	长虹	CHD32300	32	40	10000	8	20070508122104
大润发闸北	康佳	LCTM2011S	20	30	2999	8	20070508122118
家乐福万里	厦华	LC32HU25	32	10	1000	16	20070508122133
易买得银都	海信	HDP2908	29	5	3000	28	20080228091720

图 9-41　数据库所有记录数据

（2）为了查询指定的销售数据，需要退回到【查询窗体】对话框。这里查询所有商场名为“大润发春申”的销售数据，首先在查询窗口中设置商场名称为“大润发春申”，然后单击【添加】按钮，如图 9-42 所示。随后单击【查看查询集合】按钮，最后的查询结果如图 9-43 所示。

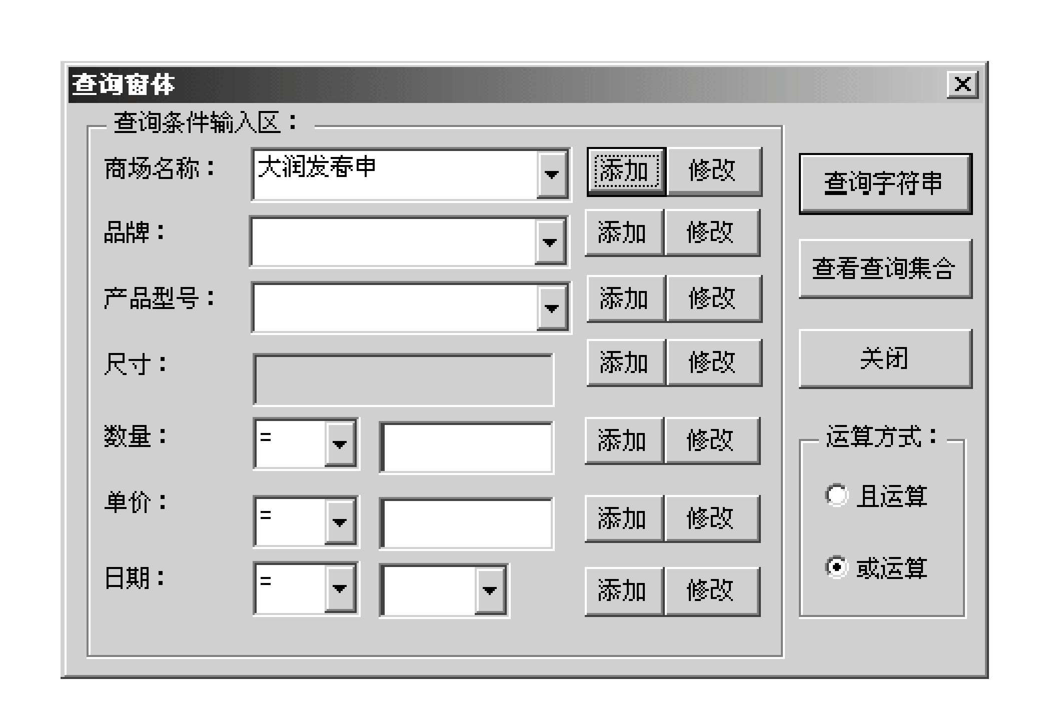


图 9-42　【查询窗体】对话框



查询结果 － 总共4条记录

商场名	品牌	产品型号	尺寸	数量	单价	日期	ID
大润发春申	LG	32LB1R	32	2	1000	8	20070508122027
大润发春申	LG	32LB1R	32	3	200	8	20070508122033
大润发春申	LG	32LB1R	32	10	20000	8	20070508122038
大润发春申	创维	21N15AA	21	2	33	29	20070529193611

图 9-43　条件查询结果

9.9.3　编辑销售数据

（1）这里做出的编辑操作即修改图 9-43 中第一条记录的产品型号。用户只需要双击该条记录即可，随后打开该条记录的编辑窗口，如图 9-44 所示。

（2）在窗口中单击【产品型号】下拉列表框，在下拉列表框中选择 42LC2RR 型号，如图 9-45 所示。【尺寸】文本框中的数据会被自动刷新。



图 9-44　销售记录编辑



图 9-45　修改销售数据

第3篇

复杂实例



第 10 章　学生座位编排系统

在日常教务工作中，往往需要针对学生的情况对学生的座位做出调整。本系统正是针对这一日常教务工作需要辅助教师完成学生座位编排工作。在该系统中，用户只需要建立学生的基本信息即可完成学生座位编排，除了自动编排方式外用户还可以自己进行后期调整工作。

10.1　系 统 概 述

在系统中完成学生座位重排工作，首先需要建立学生的资料，然后对重排后的座次行列位置以及讲台位置进行设置，最后用户可以选择系统自动排列座次或者手动调整学生座次完成排列学生座次位置工作。使用该系统完成编排座位工作的操作流程图如图 10-1 所示。

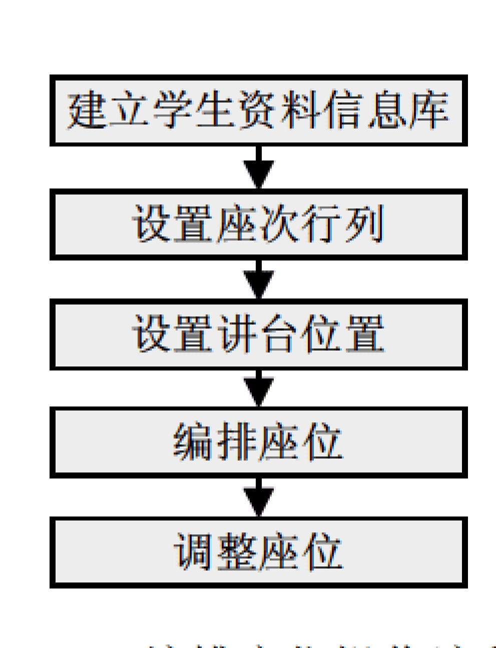
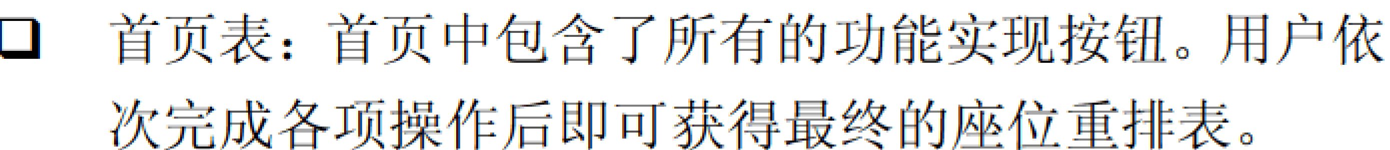


图 10-1　编排座位操作流程图

本系统一共建立了 3 个工作表、5 个窗体以及 1 个代码模块。以下是这些表、窗体和代码模块的功能简述。



- 首页表：首页中包含了所有的功能实现按钮。用户依次完成各项操作后即可获得最终的座位重排表。
- 学生表：该表保存了所有学生的资料信息。通过首页的【学生名输入】按钮可以直接跳转到该页面。
- 编排表：该表显示学生座次编排效果。在该表中，还可以对编排效果进行调整。
- 辅助输入窗口：该窗口用于辅助学生资料信息输入。在学生表中，单击【辅助输入】按钮，该窗口将被打开。
- 讲台位置窗口：该窗口用于设置座次编排表中讲台所处位置。
- 交换位置窗口：该窗口用于手动调整某两位学生的座次位置。该功能通过首页的【调整座位】按钮激发。
- 手动调整窗口：该窗口用于手动将学生编排到学生座位表中。每当一个学生被安排后，窗口中学生名列表将把该学生信息从列表中删除。
- 行列设置窗口：该窗口用于控制最终编排学生座位表的行列数。程序将把所定义的行列用边框包含起来，以区别于其他 Excel 表的单元格。
- 公用模块：该模块包含了公用变量定义、首页的跳转过程宏以及其他自定义功能过程。

10.1.1　知识点一：合并单元格

在 Excel 2007 中，为了达到对齐、美观的效果经常需要将部分单元格进行合并。通过手动操作合并单元格十分简便，首先选择需要合并的单元格，在【开始】菜单中依次选择【对齐方式】|【合并单格】命令即可。或者直接右击这些选定的单元格，然后在弹出的快捷菜单中选择【设置单元格格式】命令。用户也可以通过按 Ctrl+1 组合键打开【设置单元格格式】对话框，在弹出的对话框中选中【合并单元格】复选框（如图 10-2 所示）。

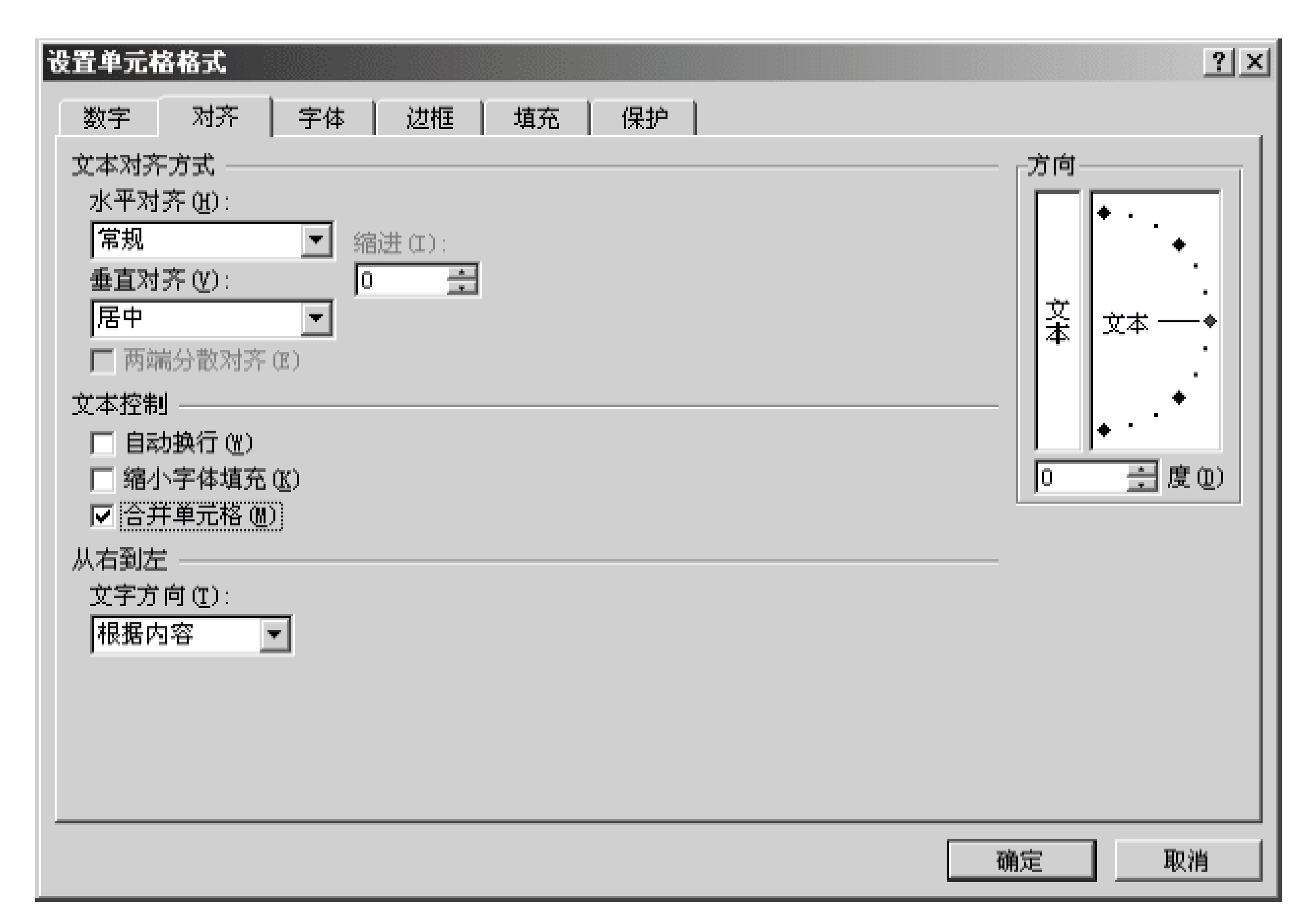


图 10-2　【设置单元格格式】对话框

要通过代码完成该操作时，只需要将这些单元格的 MergeCells 属性设置为 True 即可。设置完成后，可以通过这些单元格的 MergeArea 对象完成对合并单元格的格式及值的设置。以下是一代码示例：

```
With sheet1.Cells(1, 1)
    .Resize(2, 1).MergeCells = True                   '合并 sheet1 的 A1 和 A2 单元格
    .MergeArea.Value = "合并单元格"                    '通过 A1 访问 MergeArea 对象，并设置合并单元格值
    .MergeArea.Borders.LineStyle = xlDouble           '通过 A1 访问 MergeArea 对象，并设置合并单元格边
                                                       框样式
    .MergeArea.HorizontalAlignment = xlCenter         '设置合并单元格的水平对齐方式
    .MergeArea.VerticalAlignment = xlCenter           '设置合并单元格的垂直对齐方式
End With
```

10.1.2　知识点二：定义批注

在 Excel 2007 中定义单元格的批注，可以对部分不易懂的数据进行解释，以便于需要时查看。批注的建立在 Excel 2007 中也可以通过手动或代码建立。

手动操作建立批注的方法是：选中需要建立批注的单元格，然后右击该单元格，在弹出的快捷菜单中选择【插入批注】命令（如图 10-3 所示）后，就可以在打开的【批注】文本框

中输入批注信息（如图 10-4 所示）。

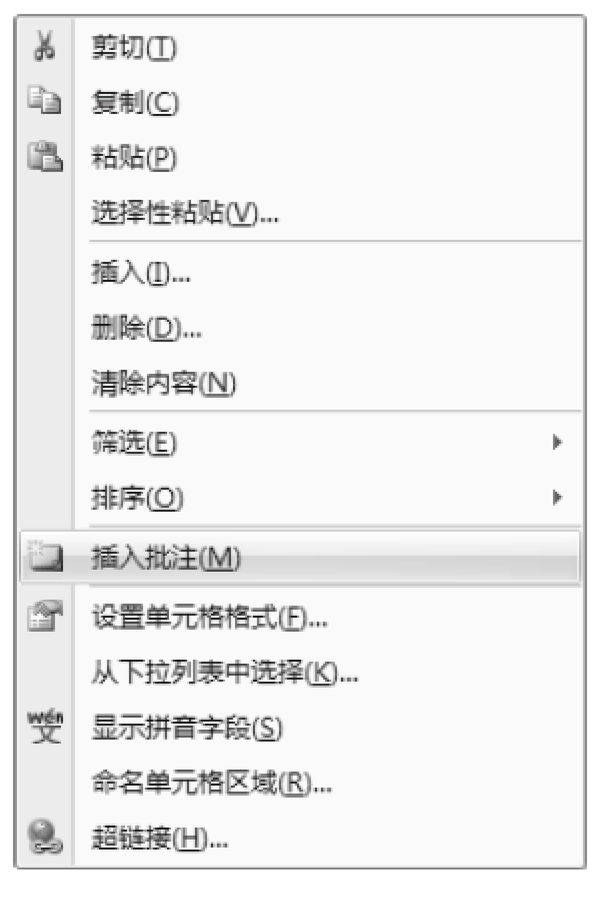


图 10-3　插入批注

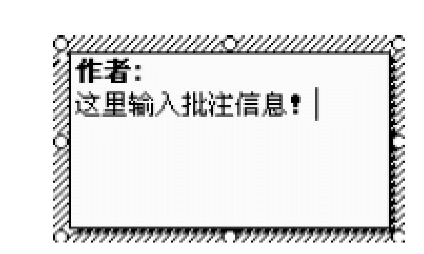


图 10-4　编辑批注信息

通过代码建立批注也十分简便。使用单元格对象的 AddComment 方法可以为该单元格添加批注。通过单元格的 Comment 对象的 Text 属性可以设置单元格批注的文字内容。以下代码为给工作表 1 中的 E5 单元格添加批注并将该批注显示出来。

```
With Worksheets("工作表 1").Range("e5").AddComment          '添加批注
    .Visible = True                                        '显示该批注
    .Text "在此输入如批注信息"                               '设置批注文字
End With
```

值得注意的一点是：批注的文字内容不能使用赋值语句完成，请仔细查看上面的代码，该代码并没有出现赋值符号“=”，而是直接将显示数据跟随在 Text 属性后面。

10.1.3　知识点三：Split 函数的使用

Split 函数是 VBA 中一个功能强大的字符串处理函数。该函数可以快速将某一字符串按照某间断字符进行分割。分割完成后，函数将各个子字符串保存到一个下标从零开始的一维数组中。下面是该函数的使用语法格式：

```
一维数组=Split(原字符串，分割字符串，子字符串数，比较方式)
```

其中，只有原字符串参数是必需的。分割字符串参数默认为空格，分割原字符串时，将以该字符串为分割间距。子字符串数定义返回子字符串的数量，默认为返回所有子字符串。比较方式定义判别子字符串时的比较方式。以下是该函数的一个使用实例。

```
myArray=Split("1/2/3/4/5","/")
```

上面的代码中，Split 函数将原字符串“1/2/3/4/5”按“/”进行分割，得到一包含 5 个元素的一维数组，然后将该数组赋给 myArray 数组变量。在立即窗口中可以输入 debug.print myArray(0)检测上面代码获得数组第一个元素的值，其显示的结果为 1。

10.2　首　页　设　计

系统概述中，对于首页表已经做了部分功能介绍。本小节将详细介绍首页的界面设计与代码设计。通过首页各个按钮和设置项目，用户可以直接完成编排工作中所有的流程步骤。由于界面中所使用的按钮是通过形状实现的，因而不存在按钮可用状态设置代码。在实际操作中，请用户注意操作的顺序。

10.2.1　首页界面设计

首页中所有的按钮都是使用形状完成的，其单击时执行的宏在公用模块中定义。除了这些按钮外，首页中还包含了一些系统设置控件。该首页的界面如图 10-5 所示。



图 10-5　学生座位编排表首页界面

首页中共包含了 6 个图形，其中一个图形作为界面框架，其余的作为按钮；还包含 1 个分组框控件以及 4 个单选按钮控件。以下是该首页中包含的这些按钮形状以及设置控件的功能简述。

- 【学生名输入】按钮：单击该按钮，将跳转到学生表中，用户在学生表中可以建立所有学生的信息。
- 【行列设置】按钮：单击该按钮，打开行列设置窗口，在该窗口中可以设置最终座次编排表的行列数。
- 【讲台位置】按钮：单击该按钮，打开讲台设置窗口，在该窗口中可以设置讲台在座次表的位置。
- 【编排座位】按钮：单击该按钮后，程序将进入座次编排工作。在这个过程中，可以选择编排座次位置或手动设置座次。
- 【调整座位】按钮：单击该按钮后，可以将已制作好座次表中的某两个学生的座次进行调换。

- 编排表显示设置：该框架控件中，包含了 4 个单选按钮。这些单选按钮用于设置座次表中显示的类型。

建立该首页界面的步骤如下：

（1）在 Excel 2007 中依次选择【插入】|【形状】|【矩形】命令。在首页空白区域单击鼠标并拖动以产生一适当大小的矩形。右击该矩形，在弹出的快捷菜单中选择【设置形状格式】命令，在打开的【设置形状格式】对话框中选择【填充】选项并选中其右侧的【纯色填充】单选按钮。随后在【颜色】下拉列表框中选择【白色，背景 1，5%】选项，如图 10-6 所示。然后再选择【阴影】选项并展开【预设】下拉列表框，在【外部】一栏中选择【右下斜偏移】选项，如图 10-7 所示。最后选择【文本框】选项，在【文字版式】分类中的【垂直对齐方式】下拉列表框中选择【顶端对齐】选项。

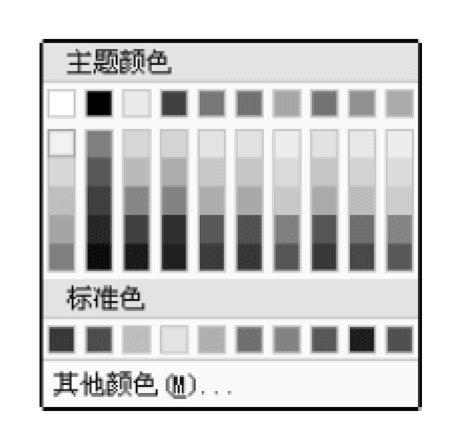


图 10-6　矩形填充颜色设置



图 10-7　矩形阴影设置

（2）右击刚创建的矩形形状，在弹出的快捷菜单中选择【编辑文字】命令。随后输入文字内容为“学生座位表制作系统”。

（3）在 Excel 2007 中依次选择【插入】|【形状】|【圆角矩形】命令。在步骤（1）和步骤（2）创建的矩形形状中插入一圆角矩形作为按钮形状。随后右击该圆角按钮，在弹出的快捷菜单中选择【设置形状格式】命令，在打开的【设置形状格式】对话框中选择【填充】选项并选中【渐变填充】单选按钮。在【预设颜色】下拉列表框并选择【麦浪滚滚】选项，如图 10-8 所示。然后选择【颜色】设置下拉列表框，在其中选择【橙色，强调文字颜色 6，淡色，80%】选项，如图 10-9 所示。

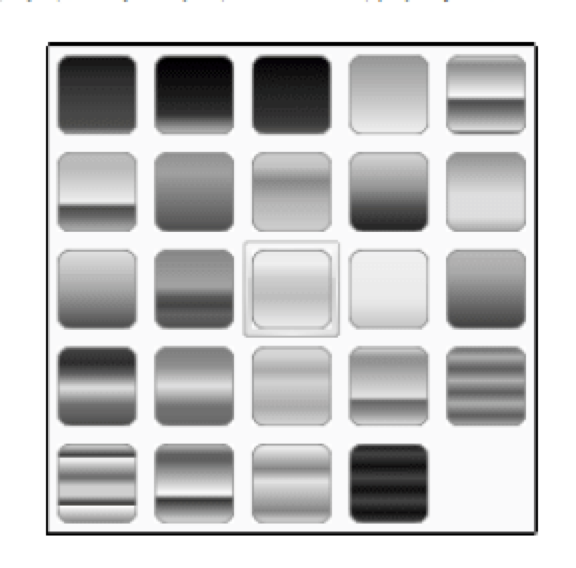


图 10-8　圆角矩形填充设置

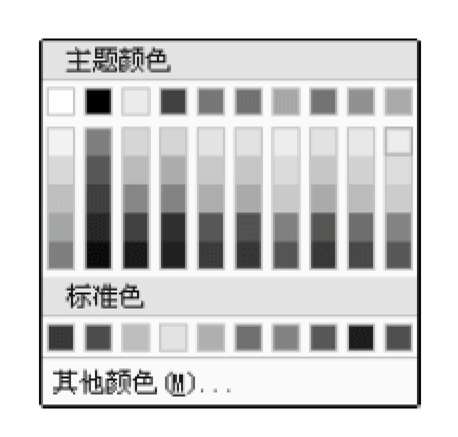


图 10-9　圆角矩形渐变颜色设置

（4）将步骤（3）建立的圆角矩形复制 4 份，按照图 10-5 所示将这 5 个圆角矩形排放在矩形形状内部。然后依次右击这 5 个圆角矩形，在弹出的快捷菜单中选择【编辑文字】命令。设置各矩形内部的文字内容依次为“学生名输入”、“行列设置”、“讲台位置”、“编排座位”和“调整座位”。

（5）右击【学生名输入】按钮并选择【指定宏】命令。在打开的【指定宏】对话框中选

择【学生输入】宏过程，如图 10-10 所示，然后单击【确定】按钮即可。与该按钮的设置方法类似，依次设置其他按钮的宏过程，其对应关系如表 10-1 所示。



图 10-10　【指定宏】对话框

表 10-1　按钮与宏名对应关系表

按　钮　名	宏　　名
行列设置	行列设置
讲台位置	讲台位置
编排座位	编排座位
调整座位	调整座位

（6）在 Excel 2007 中依次选择【开发工具】|【插入】|【表单控件】|【分组框】命令。在以上创建圆角矩形的下方插入一适当大小的分组框。随后右击该分组框，在弹出的快捷菜单中选择【编辑文字】命令，将其文字内容设置为【编排表显示设置】。

（7）在 Excel 2007 中依次选择【开发工具】|【插入】|【表单控件】|【选项按钮】命令。在以上创建分组框内依次插入 4 个选项按钮。随后依次右击这些选项按钮，在弹出的快捷菜单中选择【编辑文字】命令。依次输入文字内容为“显示姓名”、“显示性别”、“显示视力分类”和“显示评价分类”。

（8）右击其中一个选项按钮，在弹出的快捷菜单中选择【设置控件格式】命令。在随后打开的【设置控件格式】对话框中选择【控制】选项卡。将【单元格链接】设置为首页的 O1 单元格，如图 10-11 所示。



图 10-11　设置选项按钮格式



注意：Excel 2007 默认的菜单中没有开发工具菜单。要使用该菜单插入控件，需要用户手动设置。设置方法在第 4 章的知识点中已有介绍。

10.2.2　首页代码设计

首页表中并不包含任何代码，但是首页中各个跳转按钮都有其指定宏。这一部分将讲述首页中各个按钮的宏代码，这些宏过程都被保存在公共模块中。这些宏过程及其大致功能介绍如下：

- 学生输入宏过程：该过程激活学生表，以便用户在学生表中完成学生资料建立工作。
- 行列设置宏过程：打开编排表行列数设置窗口，设置后获得行列数将被保存到公共变量中。
- 讲台位置宏过程：打开讲台位置设置窗口，设置后获得的讲台位置信息被保存到公共变量中。
- 编排座位宏过程：执行编排座位操作。用户在该过程执行中途可以设置是否自动分配座次。
- 调整座位宏过程：打开调整座次窗口，并跳转到编排表中，以便用户对学生座次进行手动调整。

以上大部分按钮的宏代码都比较简单，本章将不再逐个分节介绍。但编排座位宏代码比较多，该过程将单独列出介绍。以下是其他各个宏过程的代码解释。

```
Sub 学生输入()
Worksheets("学生表").Select                    '激活学生表
End Sub

Sub 行列设置()
frm 行列设置.Show                              '显示行列设置窗口
End Sub

Sub 讲台位置()
frm 讲台位置.Show                              '显示讲台位置设置窗口
End Sub

Sub 调整座位()
Worksheets("编排表").Select                    '激活编排表
frm 交换位置.Show                              '显示交换位置窗口
End Sub
```

10.2.3　编排座位宏代码设计

编排座位宏过程按照用户设置的行列信息以及讲台位置信息完成编排座次工作。该过程的执行流程如图 10-12 所示。

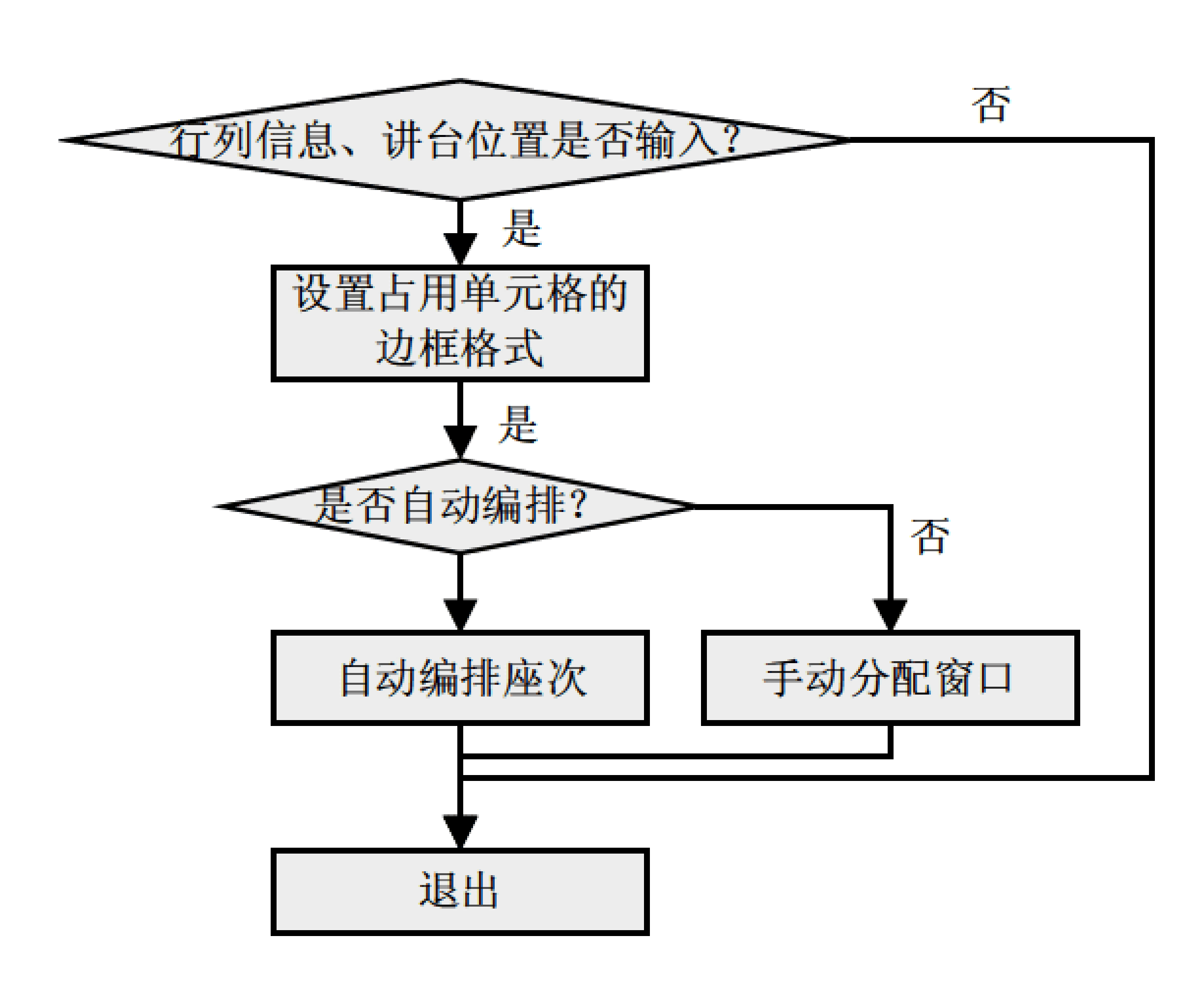


图 10-12　编排座位宏流程图

首先过程将检测行列信息与讲台位置信息是否被正确输入，否则将会退出过程，然后根据行列信息、讲台位置信息给该座次表所使用到的单元格设置边框。用户可以根据该边框划定的范围获取该座次表的大体布局，然后程序将询问是否自动分配座次，否则将会弹出手动分配窗口，用户通过该窗口可以手动分配座次。该过程的代码如下：

```
Sub 编排座位()
Dim ws As Worksheet, rg As Range, rowCount As Integer
Dim intStartRow As Integer, intStartColumn As Integer
Dim strTemp As String
Set ws = Worksheets("编排表")
ws.Cells.Clear                              '清除编排表的所有内容，以便重排格式与显示数据
'检测行数和列数信息是否被正确设置
If int 列数 <= 0 Or int 行数 <= 0 Then
    MsgBox "座位的行列数未定义！", vbInformation + vbOKOnly, "提示"      '提示行列数未定义
    Exit Sub                                '退出过程
End If
'检测讲台位置是否被设置
If Len(str 讲台位置) <= 0 Then
    MsgBox "讲台位置未定义！", vbOKOnly + vbInformation, "提示"          '提示讲台位置未定义
    Exit Sub                                '退出过程
End If
ws.Activate                                 '激活编排表
'以下设置讲台的显示位置，为了将讲台显示在座位表的中间位置，需要对讲台位置、列数的奇偶情况分
别操作
If str 讲台位置 = "左侧" Then
    If int 列数 Mod 2 = 0 Then
        With ws.Cells(int 列数 / 2, 1)
        '列数为偶数时，需要使用中间两个单元格显示讲台，以让讲台居中
            .Resize(2, 1).MergeCells = True             '合并中间两个单元格
            .MergeArea.Value = "讲台"                   '设置合并单元格的显示内容
            .MergeArea.Borders.LineStyle = xlDouble     '设置合并单元格的边框
```

```
                .MergeArea.HorizontalAlignment = xlCenter          '设置合并单元格的水平对齐方式
                .MergeArea.VerticalAlignment = xlCenter            '设置合并单元格的垂直对齐方式
            End With
            Else
            '列数为奇数时，只需要中间一个单元格，即可将讲台居中
            With ws.Cells(int 列数 / 2 + 1, 1)
                .Value = "讲台"                                    '设置讲台单元格显示内容
                .Borders.LineStyle = xlDouble                      '设置讲台单元格边框
                .HorizontalAlignment = xlCenter                    '设置讲台单元格水平对齐方式
            End With
        End If
        Set rg = ws.Cells(i, j)
Else
        If int 列数 Mod 2 = 0 Then
            With ws.Cells(1, int 列数 / 2)
                .Resize(1, 2).MergeCells = True
                .MergeArea.Value = "讲台"
                .MergeArea.Borders.LineStyle = xlDouble
                .MergeArea.HorizontalAlignment = xlCenter
                .MergeArea.VerticalAlignment = xlCenter
            End With
        Else
            With ws.Cells(1, int 列数 / 2 + 1)
                .Value = "讲台"
                .Borders.LineStyle = xlDouble
                .HorizontalAlignment = xlCenter
            End With
        End If
        Set rg = ws.Cells(j, i)
End If
'设置编排表所用行列单元格的边框
For i = 1 To int 列数
        For j = 2 To int 行数 + 1
            '讲台的位置决定了编排表的行列布局，讲台在左侧和讲台在顶部，其行列设置是互换的
            '这里需要根据讲台位置情况定义需要设置单元格的位置
            If str 讲台位置 = "左侧" Then
                Set rg = ws.Cells(i, j)
            Else
                Set rg = ws.Cells(j, i)
            End If
            rg.Borders.LineStyle = xlDouble                        '设置单元格的边框
        Next
Next
'询问是否自动分配学生或手动分配并完成分配工作
rowCount = 1
If MsgBox("是否自动分配学生？", vbQuestion + vbOKCancel, "询问") = vbOK Then
        For i = 1 To int 列数
```

```
            For j = 2 To int行数 + 1
                '根据讲台位置确定别分配单元格位置
                If str讲台位置 = "左侧" Then
                    Set rg = ws.Cells(i, j)
                Else
                    Set rg = ws.Cells(j, i)
                End If
                If rowCount <= rowsCount Then
                    '给选定单元格赋值，该值的类型由首页的设置单元格 O1 确定
                    rg = Worksheets("学生表").Cells(rowCount + 1, Worksheets("首页").Range("O1") + 1)
                    '将当前学生所有信息保存到一临时字符串变量
                    strTemp = "序号：" & Worksheets("学生表").Cells(rowCount + 1, 1) & Chr(10)
                    strTemp = strTemp & "姓名：" & Worksheets("学生表").Cells(rowCount + 1, 2) & Chr(10)
                    strTemp = strTemp & "性别：" & Worksheets("学生表").Cells(rowCount + 1, 3) & Chr(10)
                    strTemp = strTemp & "视力：" & Worksheets("学生表").Cells(rowCount + 1, 4) & Chr(10)
                    strTemp = strTemp & "评价：" & Worksheets("学生表").Cells(rowCount + 1, 5)
                    '将学生所有信息写入选定单元格的批注中
                    With rg.AddComment
                        .Visible = False
                        .Text strTemp
                    End With
                    rowCount = rowCount + 1                    '登记已自动分配学生的数量
                End If
            Next
        Next
Else
        frm手动调整.Show                                    '显示手动调整窗口
End If
Set ws = Nothing
End Sub
```

代码说明：

- 学生座次自动分配过程中，需将学生的对应信息显示在被分配单元格中。这些需要显示在单元格中的信息类型是由首页中编排表显示设置中的选项按钮确定的。这些选项一共 4 个，第一个选项按钮显示学生名被单击时，其链接的 O1 单元格的值被设置为 1。第二个选项按钮被单击时，该值为 2，依次类推。而在学生表中对应的学生名、性别、视力和评价栏被序号栏后移了一栏，因而程序中使用“Worksheets("首页").Range("O1") + 1”来定位该被分配学生的显示信息列号。
- 当用户选择了“显示学生名”外的显示设置后，编排表中显示的内容无法反映相应的学生名称等其他情况。为了便于用户及时查询被安排学生的详细情况，程序通过单元格批注保存该学生的详细情况。该批注在该单元格被单击时会显示出来，由于该显示操作属于编排表的设计部分，这里没有介绍。该部分代码请见 10.4 节具体介绍。

10.3　学生表设计

学生表主要用于完成学生信息输入工作。该表的结构并不复杂，包含了 5 列相关学生信息，还有辅助输入学生信息的【辅助输入】按钮和【二字名称对齐】按钮。以下是这两个按钮功能的简单介绍：

- 【辅助输入】按钮：该按钮被单击时，打开辅助输入学生信息窗口。该窗口只用于建立新的学生信息，没有其他的编辑、删除与查询功能。
- 【二字名称对齐】按钮：学生的名字可能只有两个汉字。在三字名称居多时，显示的编排表姓名不易对齐。该按钮将在学生名列的二字名称中添加两个空格以与三字名称对齐。

10.3.1　学生表界面设计

学生表的界面如图 10-13 所示。该表的界面没有过多的形状设计，以下将简述该表的建立步骤。



图 10-13　学生表界面设计

（1）右击首页的标签，在弹出的快捷菜单中选择【新建】命令，打开【新建】对话框。在对话框的【常用】选项卡中选择【工作表】选项，如图 10-14 所示。单击【确定】按钮插入一新工作表。随后双击该工作表的标签，修改该工作表的标签文字为“学生表”。



图 10-14　插入新表

（2）在该工作表的首行中，依次输入列标题“序号”、“学生名”、“性别”、“视力”

和“评定”。然后在 Excel 2007 中依次选择【视图】|【窗口】|【冻结窗口】|【冻结首行】命令。此时该工作表的标题行将被固定下来。

（3）复制首页的两个圆角矩形到“学生表”工作表中。然后依次右击两圆角矩形，在弹出的快捷菜单中选择【编辑文字】命令。设置两圆角矩形的文字内容分别为“辅助输入”和“二字名称对齐”。

（4）再次右击以上创建的两个圆角矩形，在弹出的快捷菜单中选择【指定宏】命令，打开【指定宏】对话框。在【指定宏】对话框中分别设置其宏过程为“辅助输入”过程和“二字名称对齐”过程。然后将首行的高度调整到与两圆角矩形形状到高度。此时圆角矩形将包含在首行之内，因而不会随表的滚动而滚动。

10.3.2　学生表代码设计

学生表中包含了表的事件代码以及相应的按钮宏代码，具体是表失去激活事件、辅助输入过程和二字名称对齐过程。这些代码都十分简单，这里不再讲述。以下是这些事件与按钮的宏代码：

```
Private Sub Worksheet_Deactivate()
'当表失去焦点时，刷新 rowsCount 公共变量，该变量记录的是已建立学生资料的数量
rowsCount = Worksheets("学生表").Cells(Rows.Count, 1).End(xlUp).Row
End Sub

Sub  辅助输入()
frm 辅助输入.Show                                          '显示辅助输入窗口
End Sub

Sub  二字名称对齐()
Dim rg As Range
rowsCount = Worksheets("学生表").Cells(Rows.Count, 1).End(xlUp).Row     '刷新 rowsCount 变量
For i = 2 To rowsCount
    Set rg = Worksheets("学生表").Range("B" & i)
    If Len(rg) = 2 Then                                    '检测学生名是否是二字名称
        rg = Left(rg, 1) & "    " & Right(rg, 1)           '二字名称中添加两空格
    End If
Next
Set rg = Nothing
End Sub
```

10.4　编排表设计

编排表实际上是一个空表，该表初始时不包含任何数据，即使存在数据，在编排操作中，程序都会自动清除数据与格式信息。但该表中仍然包含了部分代码，这些代码的用途在随后的代码设计中讲述。

10.4.1 编排表界面设计

下面给出的编排表界面是程序自动生成编排数据，该界面不是手动完成的。界面如图 10-15 所示。

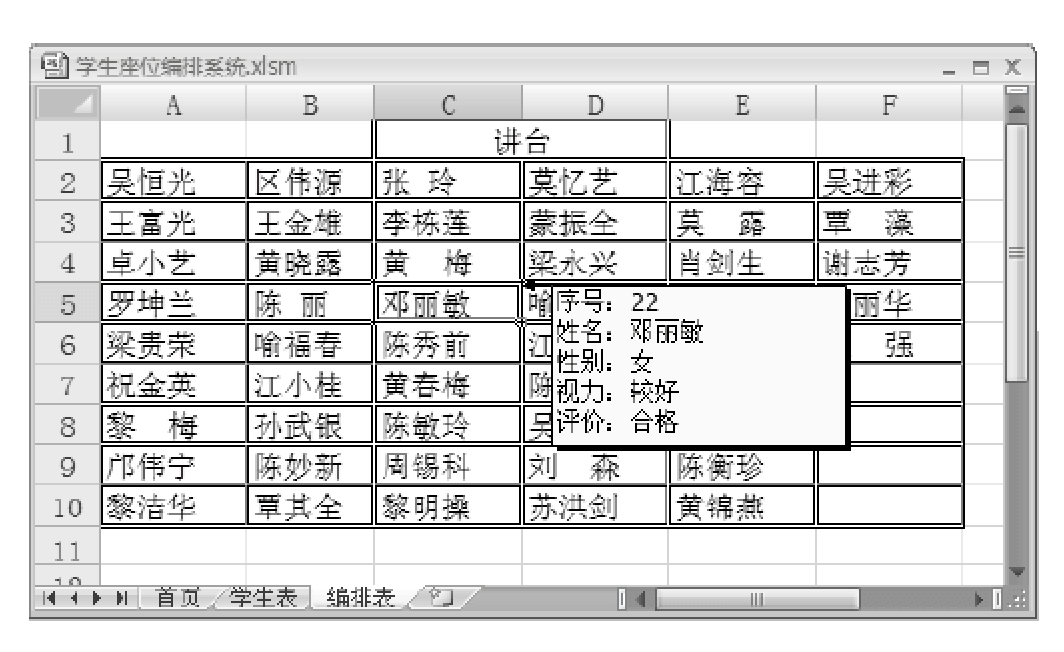


图 10-15 学生编排表界面

图中保存当前选定学生的详细信息的标签被显示出来，是因为该学生所在单元格被选择造成的。该图是使用显示学生名以及讲台在顶端设置下的编排结果。只要用户不再次单击首页中的【编排座次】按钮，该表的数据不会被清除。用户可以选择首页中其他的显示设置，来查看编排表中其他特征的分布情况，以便于调整座次。

10.4.2 编排表代码设计

在编排表中共包含了 3 个事件代码，分别是工作表激活事件、工作表失去激活事件和工作表单元格选择改变事件。这 3 个事件分别完成的功能大致描述如下：

- 工作表激活事件：当工作表被激活时，用户可能在首页中刚设置了新的显示设置项。此时需要把编排表中的所有数据重新显示出来。工作表激活事件正是完成该工作。
- 工作表失去激活事件：当工作表失去激活时，需要将所有批注项目都隐藏起来。以免在下次进入编排表时，原来被显示的批注造成干扰。
- 工作表单元格选择改变事件：当编排表中选择单元格发生变化时，需要确定该单元格是否需要显示批注项目。当需要显示批注时，应该首先隐藏原显示批注项然后显示当前选择单元格的批注。该工作由该事件完成。

```
Dim rgPrevious As Range

Private Sub Worksheet_Activate()
Dim strTemp As String, myArray() As String
Dim i As Integer
'循环所有批注项，从批注中获取需要显示在单元格中的数据，然后将该数据写入到对应单元格中
For Each cmt In Sheet3.Comments
    myArray = Split(cmt.Text, Chr(10))                    '分割批注信息，将其保存到一个自定义数组中
    strTemp = myArray(Worksheets("首页").Range("O1"))      '获取需显示信息的所有字符串数据
```

```
        i = InStr(1, strTemp, "：")                           '获取冒号的位置
        strTemp = Right(strTemp, Len(strTemp) -i)             '获取需显示到单元格的信息
        cmt.Parent.Value = strTemp                            '给单元格赋值
Next
End Sub

Private Sub Worksheet_Deactivate()
'循环所有批注，将所有批注的 Visible 属性设置为 False
For Each cmt In Sheet3.Comments
        cmt.Visible = False                                   '隐藏批注
Next
Set rgPrevious = Nothing                                      '置空 rgPrevious 单元格对象
End Sub

Private Sub Worksheet_SelectionChange(ByVal Target As Range)
On Error GoTo End_Sub
'当没有原选择单元格时，将当前选择单元格设置为原选择单元格，并将该单元格的批注显示出来
If rgPrevious Is Nothing Then
        Set rgPrevious = Target                               '为 rgPrevious 单元格对象赋值
        rgPrevious.Comment.Visible = True                     '显示选择单元格的批注
        Exit Sub
Else
        '当存在原选择单元格且该单元格不是当前选择单元格时，首先隐藏原单元格的批注，然后显示选择
         单元格批注
        If rgPrevious Is Target Then Exit Sub                 '原单元格就是当前选择单元格时，退出过程
        rgPrevious.Comment.Visible = False                    '隐藏原选择单元格批注
        Target.Comment.Visible = True                         '显示现在选择单元格批注
        Set rgPrevious = Target                               '重置 rgPrevious 单元格对象
End If
End_Sub:
End Sub
```

代码说明：

- 在代码中，定义了一个单元格对象局部变量，该变量用于记录原选择单元格。当用户在编排表中选择了一个单元格时，将会触发单元格选择改变事件（鼠标和键盘操作都会触发事件）。此时，程序将把原选择单元格的批注隐藏起来，然后显示当前选择单元格的批注，最后把当前选择单元格定义为原选择单元格。如此循环往复，完成批注的显示和隐藏工作。
- 在工作表激活事件中，需要重新设置单元格显示数据。学生的所有信息资料实际上已经保存在了该单元格的批注之中。这些批注资料将学生的所有信息项目通过 chr(10) 非打印字符连接。该非打印字符将造成换行效果，因而在批注显示时，看到的信息条目是逐行显示的。
- 通过 Split 函数按照 chr(10)字符分割后，获得的字符串信息仍然不适合显示在相应单元格上，因为其中包含了一些提示信息。但是可以根据提示信息部分的冒号位置获

取正确的显示信息数据。程序使用 InStr 函数获取冒号在字符串中的位置，然后通过 Right 函数获取的正确的显示数据。

10.5 辅助输入窗口设计

辅助输入窗口用于辅助用户在学生表中完成学生信息输入工作。该窗口只能用于进行学生信息建立操作，不包含学生资料信息编辑、删除和查询操作。该窗口的主要作用是保证输入的学生信息按行排列而不出空行数据。

10.5.1 辅助输入窗口界面设计

该窗口包含了 5 项有关学生的信息资料，分别是序号、学生名、性别、视力和评定项。其中序号是系统自动产生的，其他各个项目都需要用户自己手动建立。该窗口的界面如图 10-16 所示。

窗口中一共包含了 6 个标签控件。其中 5 个用于显示提示信息，另外 1 个用于显示自动生成的序号，该序号并不固定。窗口中包含 1 个文本框控件、3 个复合框控件和 1 个【确认】按钮。表 10-2 列出了除 5 个显示提示信息外的所有控件的控件名、控件类型和控件说明。



图 10-16　辅助输入窗口界面

表 10-2　辅助输入窗口控件列表

控件名称	控件类型	控件说明
lab 序号	标签	显示由系统自动生成的序号。该序号由学生表中最后一位学生的序号数加 1 得到
txt 学生名	文本框	该文本框中用于设置学生的学生名
comb 性别	复合框	该复合框用于设置学生性别。在窗体初始化代码中，该复合框被添加了两项“男”、“女”
comb 视力	复合框	该复合框设置学生的视力状况。复合框将学生视力分为 4 个级别。分别是“好”、“较好”、“较差”、“差”
comb 评定	复合框	该复合框设置教师对学生学习情况的评价等级。一共包含了 4 个等级“优秀”、“优良”、“合格”、“差”
btn 确认	按钮	该按钮被选择后，程序将把当前输入的学生资料填写到学生表中。其中学生名不能为空

建立该窗口的步骤如下：

（1）在 VBE 开发环境中依次选择【插入】|【用户窗体】命令。在属性窗口中将该窗体的名称属性设置为“frm 辅助输入”，如图 10-17 所示。

（2）在工具箱中选择标签控件。在窗体中连续插入 6 个标签控件。在属性窗口中将第二

个标签的名称属性设置为“lab 序号”，其他标签保持默认即可。Caption 属性依次设置为：“序号：”、“”、“学生名：”、“性别：”、“视力：”和“评定：”。注意第二个标签的 Caption 属性被设置为空，如图 10-18 所示。



图 10-17　辅助输入窗体属性设计

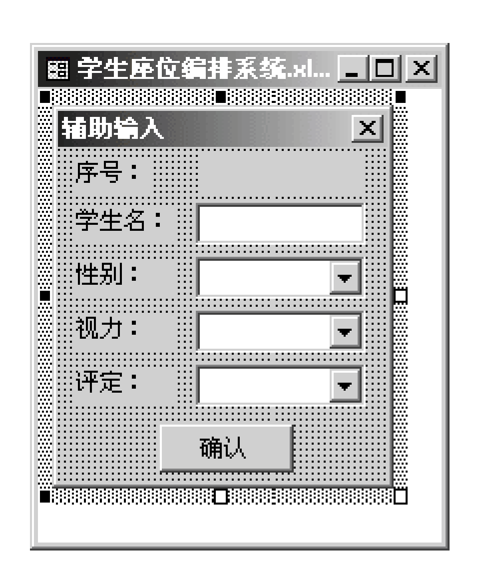


图 10-18　辅助输入窗体效果示意图

（3）在工具箱中选择文本框控件。在“学生名：”标签控件右侧插入一文本框控件。然后在属性窗口中设置该文本框的名称属性为“txt 学生名”。

（4）在工具箱中选择复合框控件。在“性别：”、“视力：”、“评定：”标签控件右侧分别插入一个复合框。随后在属性窗口中设置这些复合框的名称属性依次为“comb 性别”、“comb 视力”和“comb 评定”。Style 属性都设置为 2-fmStyleDropDownList，如图 10-19 所示。



图 10-19　设置复合框控件的 Style 属性

（5）在工具箱中选择按钮控件。然后在窗体的下方插入一个按钮。随后在属性窗口中设置该按钮的 Caption 属性为“确认”，名称属性为“btn 确认”。

10.5.2　窗口初始化代码设计

辅助输入窗口被初始化时，需要完成部分工作，包括计算序号并显示、初始化性别复合框项目、初始化视力复合框项目、初始化评定复合框项目以及将鼠标输入焦点定位到学生名输入框中。这些工作并没有执行顺序，在本程序中其执行流程如图 10-20 所示。

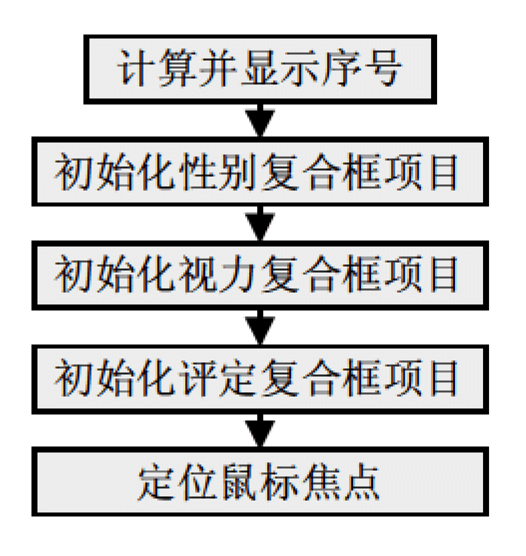


图 10-20　辅助输入窗口初始化过程流程图

以下是该窗口初始化的代码解释：

```
Private Sub UserForm_Initialize()
rowsCount = Worksheets("学生表").Cells(Rows.Count, 1).End(xlUp).Row
'计算序号并将序号显示在序号标签上
If rowscount<>1 then
    lab 序号.Caption = CStr(Worksheets("学生表").Cells(rowsCount, 1).Value + 1)
else
    lab 序号.Caption= “1”
End If
```

```
'初始化性别复合框项目
With comb 性别
    .Clear                              '清除性别复合框项目
    .AddItem "男"                       '添加第一个项目
    .AddItem "女"                       '添加第二个项目
    .Value = "男"                       '指定复合框的初始显示值
End With
'初始化视力复合框项目
With comb 视力
    .Clear                              '清除视力复合框项目
    .AddItem "好"                       '添加第一个项目
    .AddItem "较好"                     '添加第二个项目
    .AddItem "较差"                     '添加第三个项目
    .AddItem "差"                       '添加第四个项目
    .Value = "好"                       '指定复合框的初始显示值
End With
'初始化评定复合框项目
With comb 评定
    .Clear                              '清除评定复合框项目
    .AddItem "优秀"                     '添加第一个项目
    .AddItem "优良"                     '添加第二个项目
    .AddItem "合格"                     '添加第三个项目
    .AddItem "差"                       '添加第四个项目
    .Value = "优秀"                     '指定复合框的初始显示值
End With
'将鼠标输入焦点定位到学生名输入框中
txt 学生名.SetFocus
End Sub
```

代码说明：

- 系统自动产生的序号是根据学生表最后一位学生的序号获取的。但是有可能在学生表中没有任何学生的资料信息，此时没有最后一位学生的序号可以获取。为避免这种意外，程序使用一个 If 语句将这种情况的序号设置为 1。
- 对性别、视力和评定项目都采用固定输入项目是为了便于最后编排表的比较调整。如果采用了输入文本框，用户在这些项目中输入的信息资料将会十分混乱。这些混乱的信息资料在编排表中没有任何参考比较价值。这些复合框的 Style 属性被设置为 2-fmStyleDropDownList 后，复合框将不能使用键盘输入数据。

10.5.3 确认按钮单击事件代码设计

单击【确认】按钮后，程序首先需要检测用户输入的学生信息是否满足要求。在该实例中，只有学生名是必要的，其他的项目都可以不用输入。然后程序将把该学生的信息写入学生表中，并且将 rowsCount 变量累加 1 来重新记录已建立信息的学生数量。

```
Private Sub btn 确认_Click()
'检测学生名是否输入
```

```
If Len(txt 学生名.Text) <= 0 Then
    MsgBox "请输入学生名！", vbInformation + vbOKOnly, "提示"   '提示未输入学生名
End If
'向学生表中添加学生信息
With Worksheets("学生表")
    .Cells(rowsCount + 1, 1) = lab 序号.Caption                '添加序号
    .Cells(rowsCount + 1, 2) = txt 学生名.Text                 '添加学生名
    .Cells(rowsCount + 1, 3) = comb 性别.Text                  '添加性别
    .Cells(rowsCount + 1, 4) = comb 视力.Text                  '添加视力
    .Cells(rowsCount + 1, 5) = comb 评定.Text                  '添加评定
End With
rowsCount = rowsCount + 1                                      '累计已登记信息的学生数量
Unload Me                                                      '退出窗口
End Sub
```

10.6　讲台位置设置窗口设计

讲台位置设置窗口中可以设置讲台最终在编排表中的位置。通过该项设置，可以控制编排表的纵横显示方式。

10.6.1　窗口界面设计

窗口的控件数量不多，这里不再以表格的形式对各个控件进行描述。窗口包含了 1 个标签控件、1 个复合框控件和 1 个按钮。标签控件用于提示其右侧的复合框控件用于输入讲台位置。复合框控件用于输入讲台在编排表中的位置。【确认】按钮用于保存讲台位置到公共变量，该公共变量在系统制作编排表时被调用。如图 10-21 所示的是该窗口的界面。

建立该窗口的步骤比较简单，以下简述该窗口的建立步骤。

（1）在 VBE 开发环境中依次选择【插入】|【用户窗体】命令。在属性窗口中将该窗体的名称属性设置为“frm 讲台位置”，如图 10-22 所示。



图 10-21　讲台位置设置窗口界面

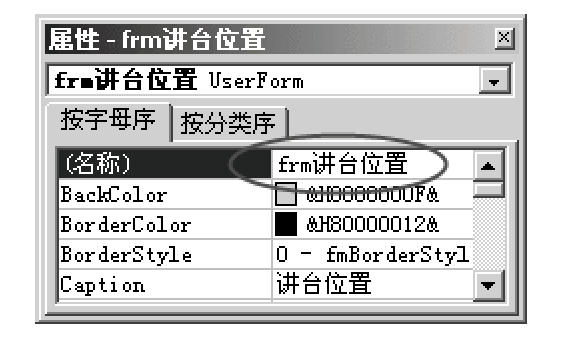


图 10-22　讲台位置窗体属性设置

（2）在工具箱中选择标签控件。在窗体的左上侧插入一个标签，在属性窗口中设置 Caption 属性为“讲台位置：”，如图 10-23 所示。

（3）在工具箱中选择复合框控件。在“讲台位置”标签控件右侧插入一个复合框。在属性窗口中设置

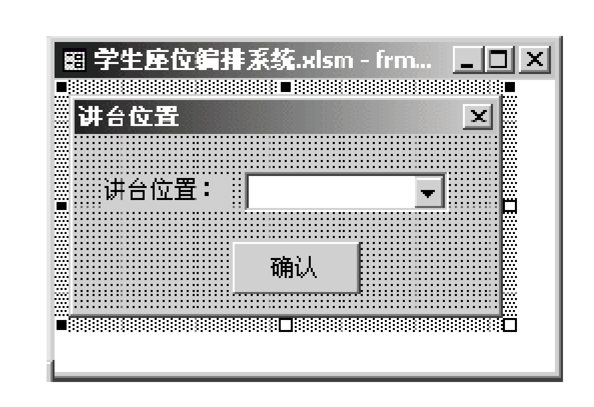


图 10-23　讲台位置窗体效果示意图



Style 属性为 2-fmStyleDropDownList。名称属性设置为“comb 讲台位置”。

（4）在工具箱中选择按钮控件。在窗口的下部插入一个按钮。在属性窗口中设置该按钮的 Caption 为“确认”，名称属性设置为“btn 确认”。

10.6.2 窗口代码设计

讲台位置窗口中的代码也不多，只有两个事件过程，分别是【确认】按钮单击事件和窗口初始化事件。【确认】按钮被单击时，程序将用户所设置的讲台位置保存到公共变量“str 讲台位置”中。在重新生成编排表时，新设置的讲台位置将发挥作用。窗口初始化时，需要为讲台位置复合框设置项目和值。以下是这两个事件过程的代码解释：

```
Private Sub btn 确认_Click()
str 讲台位置  = comb 讲台位置.Text          '保存讲台位置设置
Unload Me                                   '卸载窗口
End Sub

Private Sub UserForm_Initialize()
comb 讲台位置.AddItem "左侧"                '设置第一个复合框项目
comb 讲台位置.AddItem "顶部"                '设置第二个复合框项目
'设置讲台位置复合框的值
If Len(str 讲台位置) Then
    comb 讲台位置.Text = str 讲台位置       '当讲台位置公共变量不为空时，复合框显示该公共变量的值
Else
    comb 讲台位置.Text = "左侧"             '当讲台位置公共变量不为空时，复合框显示默认值
End If
End Sub
```

10.7 交换位置窗口设计

交换位置窗口用于在编排表中对两个学生的座次进行调整。该窗口在首页中单击【调整座位】按钮时被打开。

10.7.1 窗口界面设计

窗口中控件数量不多，包含了 2 个标签控件、2 个 RefEdit 控件和 1 个按钮控件。2 个标签控件用于提示其右侧的 RefEdit 控件的作用对象。2 个 RefEdit 控件保存被交换内容的两个单元格的位置信息。交换按钮完成交换的操作。如图 10-24 所示的是该窗口的界面效果。

建立该窗口的步骤如下：

（1）在 VBE 开发环境中依次选择【插入】|【用户窗体】命令。在属性窗口中将该窗体的名称属性设置为“frm 交换位置”，如图 10-25 所示。



图 10-24　交换位置窗口界面



图 10-25　交换位置窗体属性设置

（2）在工具箱中选择标签控件。在窗口的左上侧插入两个标签。在属性窗口中设置两标签控件的 Caption 属性依次为“交换者：”和“被交换者：”，如图 10-26 所示。

（3）在工具箱中选择 RefEdit 控件。在两标签控件右侧各插入一个 RefEdit 控件。在属性窗口中依次设置这两个 RefEdit 控件的名称属性为“Ref 交换者”和“Ref 被交换者”。

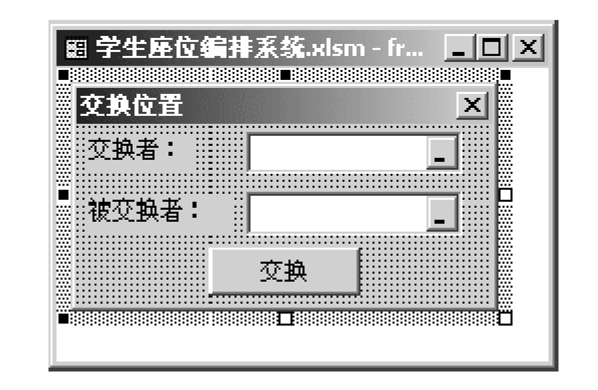


图 10-26　交换位置窗体效果示意图

（4）在工具箱中选择按钮控件。在窗口的下部插入一个按钮。在属性窗口中设置该按钮的 Caption 为“交换”，名称属性为“btn 交换”。

10.7.2　窗口代码设计

窗口中仅包含了一个交换按钮单击事件。该按钮被单击时，要完成两个工作。一方面要将选择的两个单元格的显示内容交换过来，另一方面还要将选择单元格的批注内容也交换过来。以下是该按钮单击事件的代码解释：

```
Private Sub btn 交换_Click()
Dim strTemp As String
'交换单元格显示数据
strTemp = Range(Ref 交换者.Value).Value                    '保存第一个单元格的显示内容
Range(Ref 交换者.Value) = Range(Ref 被交换者.Value)   '将第二个单元格的显示内容赋给第一个单元格
Range(Ref 被交换者.Value) = strTemp                        '第二个单元格从临时变量中获取显示数据
'交换单元格的批注信息
strTemp = Range(Ref 交换者.Value).Comment.Text        '保存第一个单元格的批注
Range(Ref 交换者.Value).Comment.Text Range(Ref 被交换者.Value).Comment.Text   '第一个单元格
                                                                                  获取批注
Range(Ref 被交换者.Value).Comment.Text strTemp        '第二个单元格从临时变量中获取批注
End Sub
```

代码说明：

- 在处理学生信息交换的两个任务的过程中，使用同一个临时字符串变量保存单元格的显示值以及单元格的批注。这并不会造成两个任务之间的混乱，因为两个任务是被依次执行的。完成交换显示值后，临时变量已经没有作用，在交换批注中修改临时变量对交换显示值没有任何影响。
- 当用户使用 RefEdit 控件获取了交换单元格后，该控件中保存了完成单元格路径。因而在获取单元格对象时直接使用了 Range(Ref 交换者.Value)，而无需再指定该单元格的表位置。

10.8 手动调整窗口设计

手动调整窗口用于在编排表中通过手动方式将所有学生编排到表中。在进行编排座位过程中，程序首先完成编排表单元格的格式设定，然后询问用户是否自动编排。当用户拒绝自动编排后，该窗口将被显示出来。

10.8.1 窗口界面设计

窗口中包含了 5 个标签控件、1 个列表框控件和 1 个按钮。5 个标签控件用于作为列表框显示内容的标题。列表框中显示了所有已登记学生的所有信息。【确认】按钮用于将当前选择的学生信息填写到编排表的选择单元格中。如图 10-27 所示的是该窗口的界面。

本窗口中的列表框控件使用了多列数据显示，通常情况下使用列表框时，只能使用一列数据值，读者可以仔细阅读本部分的控件建立和代码设计，以了解列表框多列数据显示和使用的方法。以下是该窗体的建立步骤：

（1）在 VBE 开发环境下依次选择【插入】|【用户窗体】命令。在属性窗口中设置名称属性为“frm 手动调整”，ShowModal 属性设置为 False，如图 10-28 所示。因为该窗体在使用过程中需要选择编排表中的单元格，所以该 ShowModal 属性不能为 True。



图 10-27 手动调整窗口界面



图 10-28 手动调整窗体属性设置

（2）在工具箱中选择标签控件。在窗口的顶端依次插入 5 个标签控件。在属性窗口中依次设置这些控件的 Caption 属性为“序号”、“学生名”、“性别”、“视力”和“评价”，如图 10-29 所示。

（3）在工具箱中选择文本框控件。在刚建立的标签控件下方插入一列表框。在属性窗口中设置该列表框的名称属性为“List 学生名”，ColumnCount 属性设置为 5，ColumnWidth 属性设置为“30,35,35,35,35”。ColumnCount 用于设置列表框的列数，ColumnWidth 用于设置每列的宽度。

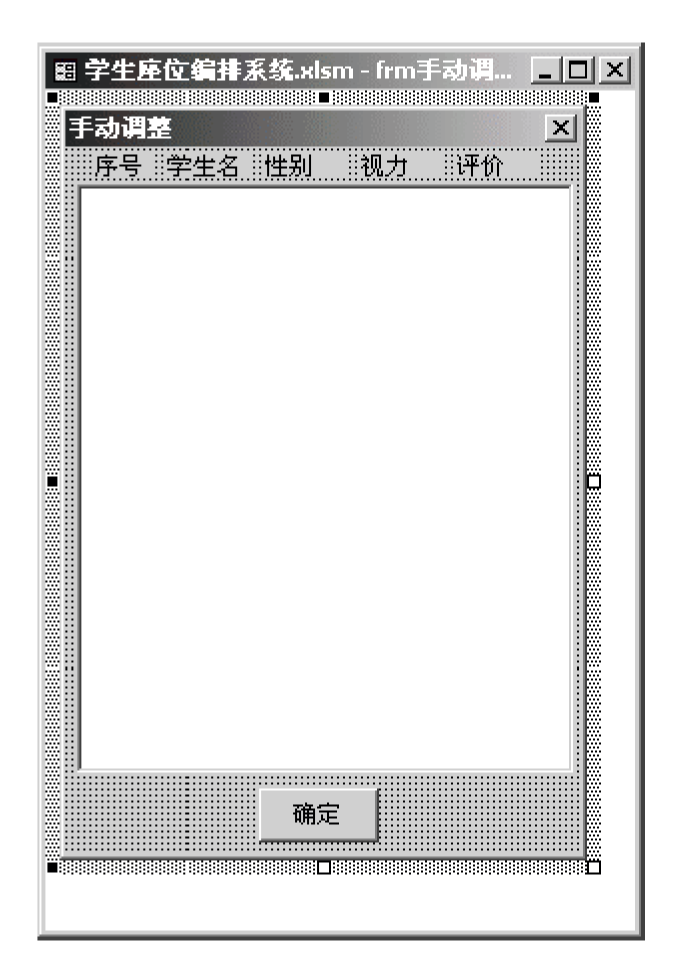


图 10-29　手动调整窗体设计效果示意图

（4）在工具箱中选择按钮控件。在窗口底部插入一按钮控件。在属性窗口中设置该按钮的 Caption 属性为“确定”，名称属性为“btn 确定”。

值得说明的是：对于列表框控件，使用多列时，仍然可以只显示一列数据出来，其他列的数据只是被隐藏起来而已。例如本例中，如果将 ColumnWidth 属性设置为“0,35,0,0,0”后，该列表框将只显示学生名列。另外多列情况下，用户还可以自定义列表框选定项目返回值所处的列，这是通过设置 BoundColumn 属性完成的。本例中使用了默认值 1 即返回序号列的值。通过该项设置开发者就可以将每一项被选择时的返回值隐藏起来，而显示出来的是用户容易理解的数据信息。

10.8.2　窗口代码设计

本窗口的代码并不长，但是其中包含了列表框控件多列数据显示时的代码设置。本小节将详细介绍该部分代码设计，以让读者能清晰了解列表框多列数据显示的代码设计。

本窗口中包含了 3 个事件过程代码，分别是窗口激活事件、窗口失去激活事件和确定按钮单击事件。这些事件过程的功能简述如下：

- 窗口激活事件：当窗口被激活时，需要重新刷新列表框中的学生信息。该事件将学生表中所有学生信息存储在一个二维数组中，然后通过使用列表框的 List 属性将该二维数组中的数据显示在列表框中。
- 窗口失去激活事件：窗口失去激活时，在编排表中被选中单元格的批注仍处于被显示状态。为了排除该批注的显示干扰，程序暂时将批注隐藏，需要显示时，用户再次选择单元格即可。
- 确定按钮单击事件：【确定】按钮被单击时，程序首先需要确认 4 个方面的事情，分别是当前被激活的表是否是编排表、激活单元格是否有学生已显示、激活单元格是否是讲台位置单元格、激活单元格的边框是否存在。这些条件的检查可以确保一件事情，即只对编排表中未被安排的编排座次单元格执行后续代码。

```
Private Sub UserForm_Activate()
Dim myArray()
rowsCount = Worksheets("学生表").Cells(Rows.Count, 1).End(xlUp).Row     '获取学生表的数据行数
ReDim myArray(rowsCount -2, 4)                                          '重新定义数组
'给数组赋值
For i = 2 To rowsCount
    myArray(i -2, 0) = Worksheets("学生表").Cells(i, 1)                 '保存序号列数据
    myArray(i -2, 1) = Worksheets("学生表").Cells(i, 2)                 '保存学生名列数据
    myArray(i -2, 2) = Worksheets("学生表").Cells(i, 3)                 '保存性别列数据
    myArray(i -2, 3) = Worksheets("学生表").Cells(i, 4)                 '保存视力列数据
    myArray(i -2, 4) = Worksheets("学生表").Cells(i, 5)                 '保存评价列数据
Next
'显示数组数据到列表框
With List 学生名
    .Clear                                                              '清除列表框数据项目
    .List() = myArray                                                   '给列表框赋值
End With
End Sub

Private Sub UserForm_Deactivate()
'循环编排表中所有的批注项目，将所有批注项目的可见属性设置为 False
For Each cmt In Sheet3.Comments
    cmt.Visible = False                                                 '隐藏批注
Next
End Sub

Private Sub btn 确定_Click()
Dim i As Integer, strTemp As String
If ActiveSheet.Name = "编排表" And Len(ActiveCell) = 0 _
    And ActiveCell <> "讲台" And ActiveCell.Borders.LineStyle = xlDouble Then
    '当编排表中的选定单元格是未被编排的座位单元格时，执行以下语句
    i = List 学生名.Value                                               '获取列表框选定项目的序号列数据
    List 学生名.RemoveItem List 学生名.ListIndex                        '移除列表框的选定项目
    '给选定单元格赋值并标记单元格批注
    ActiveCell = Worksheets("学生表").Cells(i + 1, Worksheets("首页").Range("O1") + 1)    '赋值
    strTemp = "序号：" & Worksheets("学生表").Cells(i + 1, 1) & Chr(10)               '序号段字符串
    strTemp = strTemp & "姓名：" & Worksheets("学生表").Cells(i + 1, 2) & Chr(10)     '连接学生名段
                                                                                      字符串
    strTemp = strTemp & "性别：" & Worksheets("学生表").Cells(i + 1, 3) & Chr(10)     '连接性别段字
                                                                                      符串
    strTemp = strTemp & "视力：" & Worksheets("学生表").Cells(i + 1, 4) & Chr(10)     '连接视力段字
                                                                                      符串
    strTemp = strTemp & "评价：" & Worksheets("学生表").Cells(i + 1, 5)               '连接评价段字
                                                                                      符串
    With ActiveCell.AddComment
        .Visible = False                                                '隐藏批注
        .Text strTemp                                                   '将批注字符串写入单元格批注中
    End With
End If
End Sub
```

代码说明：

- 使用列表框控件建立多列数据显示时，首先需要将这些数据保存在一个二维数组中。代码中没有使用 Option Base 1 语句来定义数组的起始索引号为 1，因而程序中定义二维数组时使用了 ReDim myArray(rowsCount −2, 4)语句。该数组包含元素实际有 rowsCount−1 行，5 列。
- 将所有数据保存到数据后，通过列表框的 List 属性可以使列表框获取前面二维数组的数据。当用户选择列表框中某一项时，该列表框的返回值将是被绑定列的数值。该绑定列的位置由 BoundColumn 属性决定。
- 当需要移除列表框中的项目时，只需要调用列表框的 RemoveItem 方法即可；该方法接受选定移除项目的索引顺序号的参数。本例中需要移除被用户选中的项目，此时可以通过列表框的 ListIndex 属性获取该被选中项目的索引号。

10.9　行列设置窗口设计

行列设置窗口用于设置座次编排表的行列数。在首页单击【编排座次】按钮后，程序将首先根据行列用户设置的行列数和讲台位置设置单元格的边框。

10.9.1　窗口界面设计

行列设置窗口的控件数量不多，包含 3 个标签控件、2 个文本框和和 1 个确认按钮。其中 1 个标签用于显示已建立信息的学生数量，其他 2 个标签用于显示提示信息。2 个文本框控件分别用于输入行数和列数。确认按钮用于保存用户设置行列数到公共变量，该公共变量在编排座次过程中被调用。图 10-30 是该窗口的界面。

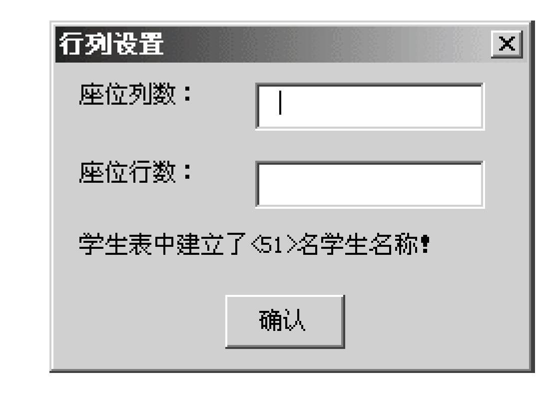


图 10-30　行列设置窗口界面

建立该窗口的步骤如下：

（1）在 VBE 开发环境下依次选择【插入】|【用户窗体】命令。在属性窗口中设置窗口的名称属性为“frm 行列设置”，如图 10-31 所示。

（2）在工具箱中选择标签控件。在窗体中连续插入 3 个标签。在属性窗口中设置前两个标签的 Caption 属性依次为“座位列数：”和“座位行数：。随后将第 3 个标签的名称属性设置为“lab 提示”，如图 10-32 所示。



图 10-31　行列设置窗体属性设置

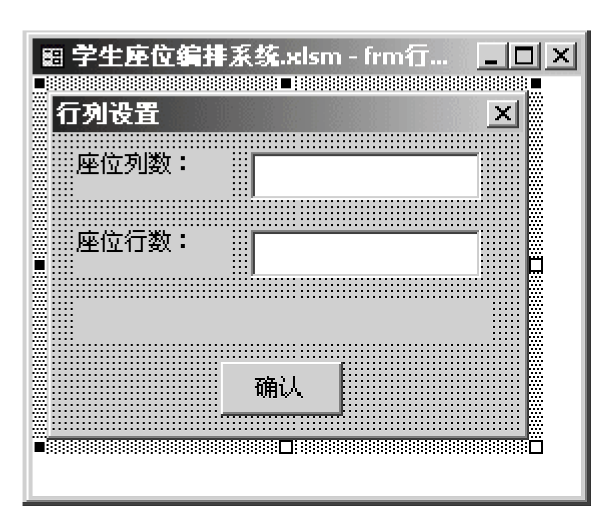


图 10-32　行列设置窗体设计效果示意图

（3）在工具箱中选择文本框控件。在窗口中连续插入两个文本框。在属性窗口中依次设置这两个文本框控件的名称属性为“txt 列数”和“txt 行数”。

（4）在工具箱中选择按钮控件。在窗口底部插入一个按钮。在属性窗口中设置按钮的 Caption 属性为“确认”，名称属性为“btn 确认”。

10.9.2 窗口代码设计

在行列设置窗口中，当用户输入了某一个数值后，程序会根据当前的学生总数自动来确认另外一个数值。当然用户可以根据该数值自己调整，但是不允许输入数值小于自动确认的数值，因为自动确认数值已经是容纳所有学生的最小数。

该窗口过共包含了 4 个事件过程代码，分别是窗口初始化事件、列数文本框更新事件、行数文本框更新事件和确认按钮单击事件。这 4 个事件过程的功能描述如下：

- ❑ 窗口初始化事件：窗口初始化时，需要获取已登记信息的学生数量，并将该数量显示在提示标签上。
- ❑ 列数文本框更新事件：列数输入数据发生变化时，程序需要根据用户输入列数和总登记学生数量确定一个最小行数，并将该最小数显示在行数文本框中。
- ❑ 行数文本框更新事件：行数输入数据发生变化时，程序需要根据用户输入行数和总登记学生数量确定一个最小列数，并将该最小数显示在列数文本框中。
- ❑ 确认按钮单击事件：单击【确认】按钮时，程序将把用户设置的行列数保存到公共变量中，以便程序在编排座次过程中调用。

以下是这些事件过程的代码解释：

```
Private Sub UserForm_Initialize()
rowsCount = Worksheets("学生表").Cells(Rows.Count, 1).End(xlUp).Row    '获取学生表总行数
lab 提示.Caption = "学生表中建立了<" & rowsCount -1 & ">名学生名称！"    '显示登记信息的学生数量
txt 列数.SetFocus                              '输入焦点定位到列数文本框
End Sub

Private Sub txt 列数_AfterUpdate()
Dim i As Integer, j As Integer
i = txt 列数.Value                             '记录列数数据
'计算最小行数
j = (rowsCount -1)/I                           '获取登记学生数量与列数相除的商数
If i * j < rowsCount -1Then
    txt 行数.Text = j + 1                       '商数与列数乘积小于学生数时，最小行数为商数加 1
Else
    txt 行数.Text = j                           '商数与列数乘积大于或等于学生数时，最小行数为商
End If
End Sub

Private Sub txt 行数_AfterUpdate()
Dim i As Integer, j As Integer
```

```
i = txt 行数.Value                        '记录行数数据
'计算最小列数
j = (rowsCount -1)/ I                     '获取登记学生数量与行数相除的商数
If i * j < rowsCount-1 Then
    txt 列数.Text = j + 1                 '商数与行数乘积小于学生数时，最小列数为商数加 1
Else
    txt 列数.Text = j                     '商数与行数乘积大于或等于学生数时，最小列数为商
End If
End Sub

Private Sub btn 确认_Click()
'检测行列数输入是否正确
If CInt(txt 列数.Text) * CInt(txt 行数.Text) < rowsCount -1 Then
    MsgBox "行列数确定的总人数小于实际人数，请重新输入！", vbInformation + vbOKOnly, "行列数
错误"
    txt 列数.Text = ""                         '清空列数文本框
    txt 行数.Text = ""                         '清空行数文本框
    txt 列数.SetFocus                          '定位输入焦点到列数文本框
    Exit Sub                                   '退出过程
End If
'输入正确时，保存行列数到公共变量
int 列数  = txt 列数.Text                      '保存列数
int 行数  = txt 行数.Text                      '保存行数
Unload Me
End Sub
```

10.10　系 统 测 试

本章节展示了使用系统为一个实际班级学生分配座次的过程。实例中使用到的学生一共 50 名，这里假设用户已经建立了这些学生的信息。测试中实际需要完成的工作非常少，用户只需要设置讲台位置和行列数就可以轻松完成座次的自动排列。以下将把测试内容分为座次编排设置与自动排列座次、调整座次两个部分。

10.10.1　座次编排设置与自动排列座次

（1）在首页单击【行列设置】按钮，系统将弹出【行列设置】对话框。这里将学生座次分为 6 列 9 行，此处只需要输入列数 6 即可。程序将会自动计算最恰当的行数，当然用户也可以输入行数，程序会自动计算最恰当的列数。设置完成后单击【确认】即可，如图 10-33 所示。

（2）在首页单击【讲台位置】按钮，系统弹出【讲台位置】对话框。这里设置讲台位置处于编排表的顶端，因而选择顶部即可，如图 10-34 所示。

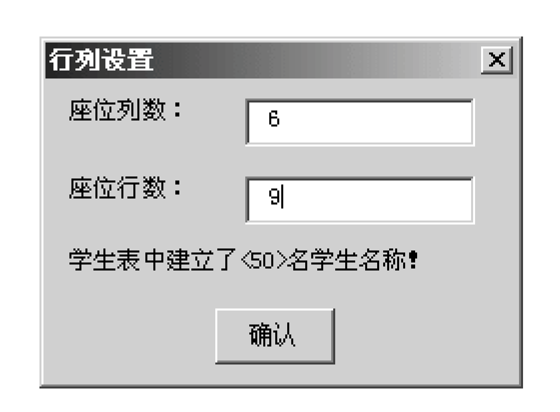


图 10-33　【行列设置】对话框



图 10-34　【讲台位置】对话框

（3）重新回到首页，单击【编排座位】按钮，随后弹出是否自动分配座次对话框，这里单击【确定】按钮让系统自动分配座次，如图 10-35 所示。分配完成后其结果如图 10-36 所示。

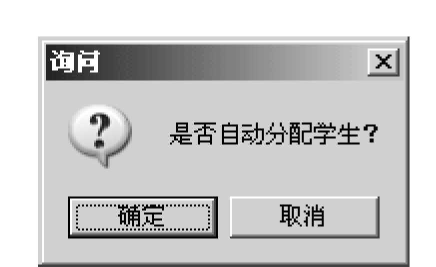


图 10-35　【是否自动分配座次】对话框



	A	B	C	D	E	F
1			讲台			
2	吴恒光	区伟源	张 玲	莫忆艺	江海容	吴进彩
3	王富光	王金雄	李栋莲	蒙振全	莫 露	覃 藻
4	卓小艺	黄晓露	黄 梅	梁永兴	肖剑生	谢志芳
5	罗坤兰	陈 丽	邓丽敏	喻福胜	韦春明	陈丽华
6	梁贵荣	喻福春	陈秀前	江河英	欧艳群	赵 强
7	祝金英	江小桂	黄春梅	陈振昌	黄紫荧	
8	黎 梅	孙武银	陈敏玲	吴柱培	刘 洁	
9	邝伟宁	陈妙新	周锡科	刘 森	陈衡珍	
10	黎洁华	覃其全	黎明操	苏洪剑	黄锦燕	

图 10-36　自动学生座位编排结果

10.10.2　调整座次

以前面的自动编排结果为基础，对 A2 和 B2 编排的学生座次进行调整。

（1）在首页单击【调整座位】按钮，系统将会自动激活编排表工作表，并且打开【交换位置】对话框，如图 10-37 所示。

（2）单击【交换位置】对话框中的交换者 RefEdit 控件，在编排表中选择 A2 单元格，如图 10-38 所示。在【交换位置】对话框中单击被交换者 RefEdit 控件，并在编排表中选择 B2 单元格，如图 10-39 所示。最终设置的结果如图 10-40 所示。最后单击【交换】按钮即完成座位调整。



图 10-37　【交换位置】对话框

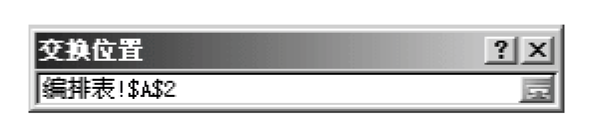


图 10-38　设置交换者

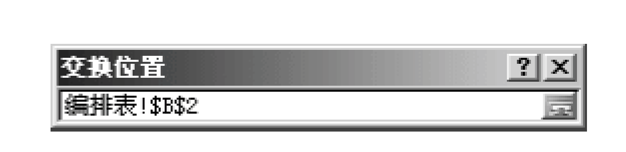


图 10-39　设置被交换者



图 10-40　交换设置结果

第 11 章　合同管理系统

合同是企业的命脉，对签订的合同进行科学管理，是一份十分重要的工作。合同涵盖了企业日常运营所有的方面，包括销售合同、购买合同、设计合同和劳务合同等。只有对这些合同采取积极而有效的管理，企业才能掌握合同的具体执行情况，才能在合同的执行过程中掌握主动权。

本实例以 Access 数据库为后台数据库。通过本实例，用户可以了解 ADO 数据库对象的使用。在前面部分章节，逐一介绍了 DAO 读写 Excel 数据库、读写 Access 数据库文件的实例。用户可以结合这些内容，在自己的应用开发中选择适用的开发方式。

11.1　系 统 概 论

合同信息管理系统共包含 4 个功能模块，即用户登录及权限管理、合同基本信息管理、合同收费信息管理和合同资料查询与导出。它们分别还包含自己的独立子功能模块，以完成各自不同的任务。该系统的功能模块结构图如图 11-1 所示。

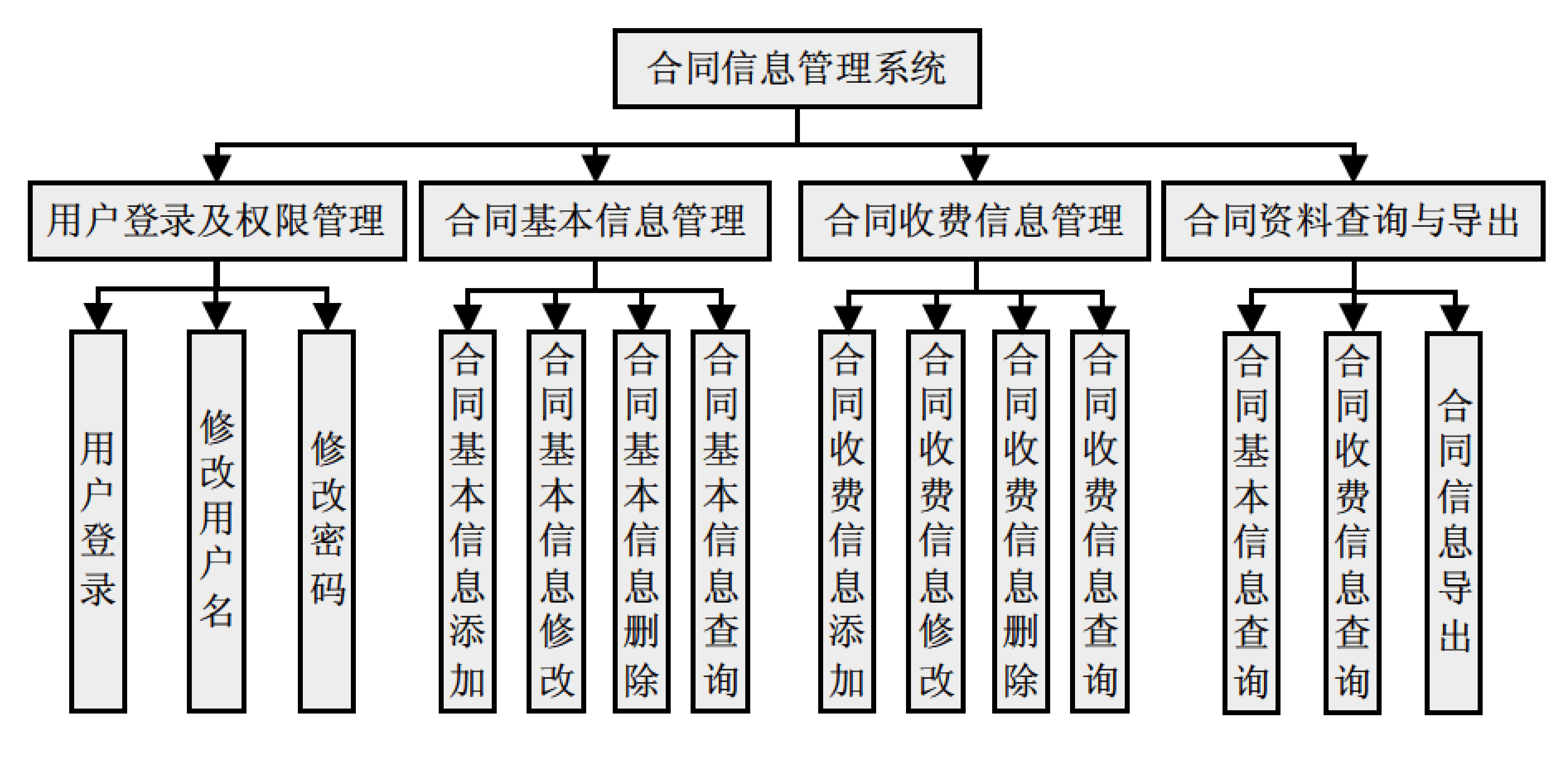


图 11-1　合同信息管理系统功能模块结构图

各个模块的功能描述如下。

- 用户登录及权限管理模块：用于用户登录、修改用户名或修改用户密码。
- 合同基本信息管理模块：用于完成对合同基本信息资料的添加、修改和删除等基本操作。在修改和删除合同基本信息资料时，首先需要查询出要修改或删除的合同基本信息。
- 合同收费信息管理模块：用于完成对合同收费信息资料的添加、修改和删除等基本

操作。在修改和删除合同收费信息资料时，首先需要查询出要修改或删除的合同收费信息。

- 合同资料查询与导出模块：用于完成对合同基本信息资料和合同收费信息资料的查询和导出等操作。查询出的结果既可以在窗体上浏览查看，也可以将查询出的资料保存到一个新工作簿中，以便于进一步进行处理。

11.1.1　知识点一：工作表的可见性

对于 Excel 2007 文件中的每个工作表都有一个可见性属性。在通常情况下，新建立的工作表的可见性属性被默认设置为可见（-1-xlSheetVisible）。该属性一共有 3 个可设置值，分别是-1-xlSheetVisible、0-xlSheetHidden 和 1-xlSheetVeryHidden。这 3 个设置值的意义分别是：

- xlSheetVisible：工作表可见。
- xlSheetHidden：工作表被隐藏。使用该隐藏方式隐藏的工作表可以通过菜单取消该工作表的隐藏。隐藏或取消隐藏的方法见知识点二。
- xlSheetVeryHidden：工作表被深度隐藏。使用该隐藏方式隐藏的工作表只能在 VBE 环境中取消隐藏。

要修改工作表的可见性，既可以在 Excel 2007 界面上修改，也可以在 VBE 编辑环境下修改，但是在 VBE 环境下才可以设置深度隐藏。下面两个知识点分别介绍各自的操作。

11.1.2　知识点二：隐藏或取消隐藏表

通过菜单方式隐藏和取消隐藏表的方法是：选择【开始】菜单栏，再选择单元格功能块中的【格式】选项，在【可见性】一栏选择【隐藏或取消隐藏】命令，将会展开一个二级菜单，该菜单如图 11-2 所示。该图中【取消隐藏工作表】显示为灰色不可用状态。因为工作簿只有两个表，一个表的可见性被设置为 xlSheetVeryHidden，该表不是普通的隐藏表。而 Excel 2007 工作簿至少要保证一个表不被隐藏。此时该工作簿的隐藏表列表中没有隐藏表，所以该项为不可用。

选择【隐藏工作表】选项时，当前被激活的工作表的可见性被设置为 xlSheetHidden。当再选择【取消隐藏工作表】选项时，在弹出的【取消隐藏】对话框中会看到所有当前工作簿中被隐藏的工作表的明细列表（如图 11-3 所示）。从中选择需要取消隐藏的表后，单击【确定】按钮即可取消该表的隐藏。



图 11-2　隐藏或取消隐藏二级菜单



图 11-3　【取消隐藏】对话框

11.1.3　知识点三：设置或取消深度隐藏

设置为深度隐藏后，该隐藏表将不会出现在【取消隐藏】对话框中。要查看这样的表的数据或格式，只有进入 VBE 环境，然后修改该表的可见性属性。修改的方法如下：

（1）在 Excel 2007 界面中按 Alt+F11 组合键进入 VBE 环境。

（2）在【工程资源管理器】中找到该深度隐藏的表，选择该工作表。此时属性面板中显示的即为该表的所有属性。

（3）找到 Visible 属性，将该属性设置为-1-xlSheetVisible。

（4）设置该表为深度隐藏表的步骤与上述步骤一致，只是步骤（3）设置的 Visible 属性为 1-xlSheetVeryHidden。

在 VBA 代码中也可以对工作表的可见属性进行设置。设置工作表可见性的语法格式如下，其中可见性参数值即为知识点一说到的 3 个系统变量。

```
工作表对象.visible=可见性参数值
```

11.1.4　知识点四：保护工作表与撤销保护

在设计工作簿时，设计者可能希望用户在工作表中只能执行某些固定的操作。例如只允许用户选择工作表中的单元格而不能进行任何编辑操作，此时需要使用工作表保护功能实现。

首先激活需要保护的工作表，然后在 Excel 2007 菜单中依次选择【审阅】|【保护工作表】命令，此时会打开【保护工作表】对话框（如图 11-4 所示）。此处可以设置是否保护工作表及锁定的单元格内容、保护工作表的密码以及用户允许的操作。设置完成后单击【确定】按钮。如果设置了密码，此时会弹出一个对话框（如图 11-5 所示）要求重新确认密码。再次输入密码后单击【确定】按钮即可。

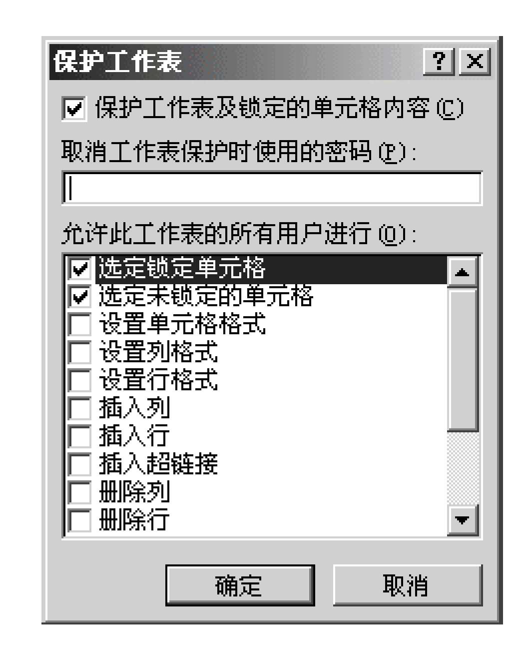


图 11-4　【保护工作表】对话框

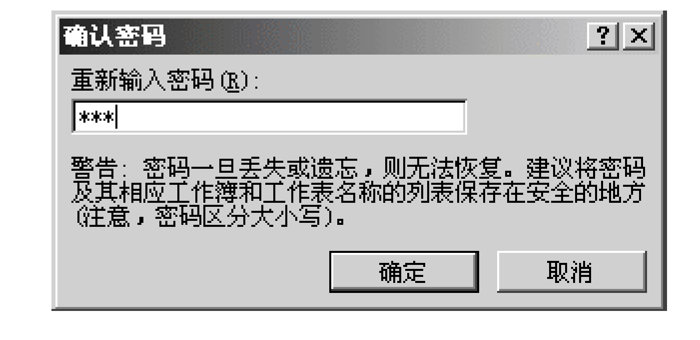


图 11-5　确认保护工作表密码

要取消工作表保护，只需要再次在 Excel 2007 中依次选择相应菜单。此时【保护工作表】菜单已经变成了【撤销工作表保护】菜单。选择该菜单，然后输入保护密码即可解除工作表的保护状态。

保护工作表与撤销保护工作表也可以通过代码实现。这两个功能分别使用工作表对象的

Protect 方法和 UnProtect 方法实现。Protect 方法的参数十分复杂，这里不再列出其语法格式。而 UnProtect 方法只接受一个可选参数 Password，该参数是保护工作表的密码。

11.2 数据表设计

在本系统中大部分数据都被保存到了 Access 数据库文件中。使用 Access 2007 打开本系统的数据库文件“合同管理.mdb”后，可以找到 5 个数据表。它们分别是部门信息表、合同基本信息表、合同类别信息表、合同收费信息表和收费类别表。这些表的功能描述如下：

- 部门信息表：用于保存企业签订合同的部门名称。
- 合同基本信息表：用于保存合同的基本信息数据。
- 合同类别信息表：保存合同类别信息数据。该信息项目将合同的性质大致做了划分，比如设计合同、承包合同、销售合同等。
- 合同收费信息表：保存合同的收费信息数据。
- 收费类别表：保存合同收费的类别数据。该信息表定义收费的性质，比如定金、预付款等。

在前面章节的实例中已经讲述过如何在 Access 中建立表以及设置表的字段结构。这里只将数据库中所有表的字段信息以表格的形式体现出来。读者在设计该数据库时，可以参考表 11-1~表 11-4 对数据库表的字段进行设置。

表 11-1　部门信息表字段设计

字 段 名 称	数 据 类 型	字 段 长 度	是否允许为空
部门名称	文本	20	否

表 11-2　合同基本信息表字段统计

字 段 名 称	数 据 类 型	字 段 长 度	是否允许为空
合同号	文本	20	否
项目名称	文本	30	否
委托单位	文本	50	否
联系人	文本	10	否
联系电话	文本	20	否
签订日期	日期		否
签订人	文本	10	否
签订部门	文本	20	否
合同起始日期	日期		否
合同终止日期	日期		否
合同金额	货币		否
合同类别	文本	20	否
备注	文本	50	是

表 11-3　合同类别信息表字段设计

字 段 名 称	数 据 类 型	字 段 长 度	是否允许为空
合同类别	文本	20	否

表 11-4　合同收费信息表字段统计

字 段 名 称	数 据 类 型	字 段 长 度	是否允许为空
合同号	文本	20	否
收费类别	文本	20	否
收费日期	日期		否
收费金额	文本		否
备注	文本	50	是

11.3　首 页 设 计

本实例通过一个首页界面将各种功能串联在一起，通过首页可以快速访问各个子功能模块。首页的界面被放置在一个工作表中。当工作簿开启时，直接进入该界面。

11.3.1　首页界面设计

首页的界面和以往章节的首页界面风格大体一致。在界面里包含了 7 个图形，其中最外围的一个图形作为边框，其他 6 个都作为按钮，如图 11-6 所示。



图 11-6　合同管理系统首页界面

建立该首页界面的步骤如下：

（1）在 Excel 2007 中依次选择【插入】|【插图】|【形状】命令。然后在矩形形状区选择第一个矩形形状。随后在首页的工作区中单击鼠标并拖动以产生适当大小的界面外边框。

（2）右击外边框，在弹出的快捷菜单中选择【设置形状格式】命令，打开【设置形状格式】对话框。在该对话框中选择【阴影】选项并在【预设】下拉列表框的“外部”分类中选择【右下斜偏移】选项，设置效果如图 11-7 所示。然后选择【文本框】项目。将“文本版式”分类中的【垂直对齐方式】设置为【顶端对齐】。

（3）选中以上创建的外边框，随后依次选择【开始】|【字体】命令。在【字体】选项卡分组中设置字体为“华文彩云”，字号为 20。设置效果如图 11-8 所示。再次右击该外边框，在弹出的快捷菜单中选择【编辑文字】命令，然后输入文字内容为“合同管理系统”。

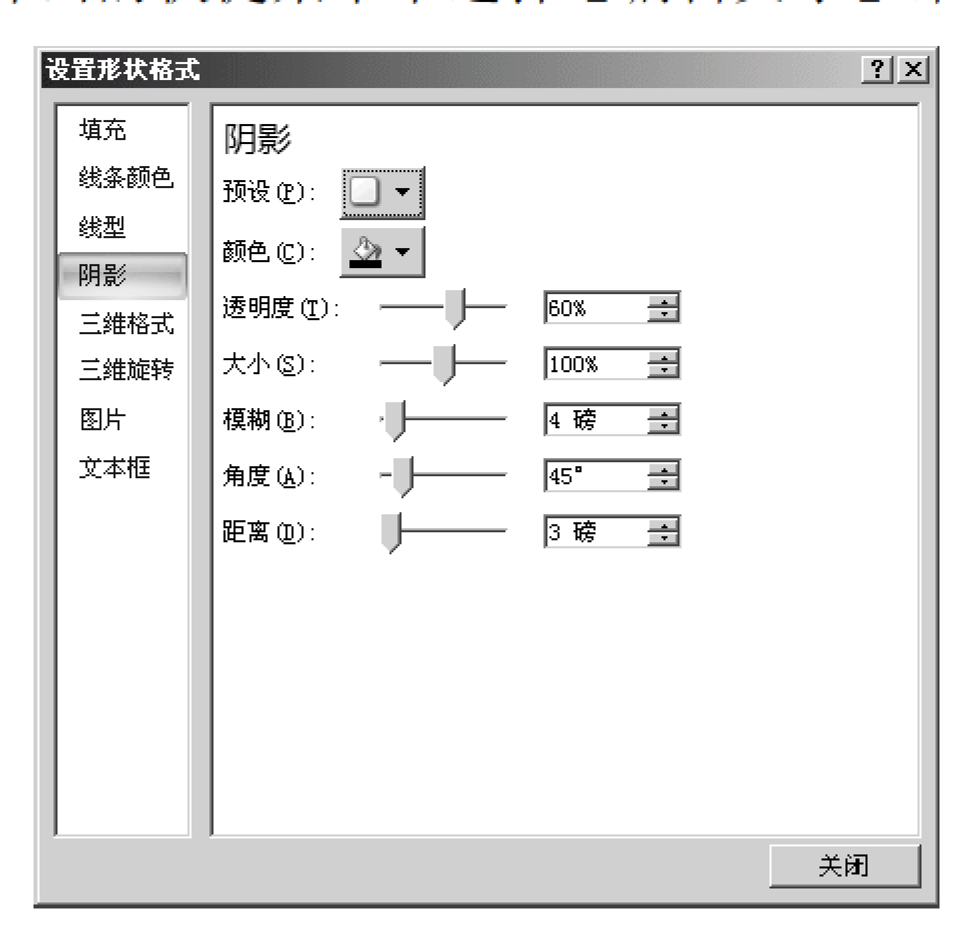


图 11-7　设置形状阴影



图 11-8　文字格式设置

（4）和前面插入矩形形状操作步骤相似，在 Excel 2007 中依次选择【插入】|【插图】|【形状】命令。然后选择矩形形状区域第二个图形——圆角矩形。随后在界面外边框内创建适当大小的圆角矩形。

（5）右击该圆角矩形，在弹出的快捷菜单中选择【设置形状格式】菜单项。在打开的【设置形状格式】对话框中将其【阴影】设置为与外边框一致。然后选择【填充】项目并选中【渐变填充】单选按钮，【类型】选择【线性】，【颜色】设置为【橙黄色】。设置的效果如图 11-9 所示。

（6）将步骤（5）创建的圆角矩形复制 5 份，调整好各个圆角矩形间的间距。然后依次右击各个圆角矩形，在弹出的快捷菜单中选择【编辑文字】命令。设置文字内容依次为“登录系统”、“修改用户名”、“修改密码”、“合同基本信息管理”、“合同收费信息管理”和“合同信息查询与导出”。

（7）为按钮添加宏。本系统的首页界面上的按钮被单击后将开启相应的窗口，这个动作是由相应的宏过程实现的，将这些宏指定到相应的按钮即可。这里只以【登录系统】按钮为例加以说明。首先右击该按钮，在弹出的快捷菜单中选择【指定宏】命令。在【指定宏】窗口的列表框中选择【登录系统】宏过程，随后单击【确定】按钮即可。效果如图 11-10 所示。各个按钮的宏代码请见后续代码设计章节。

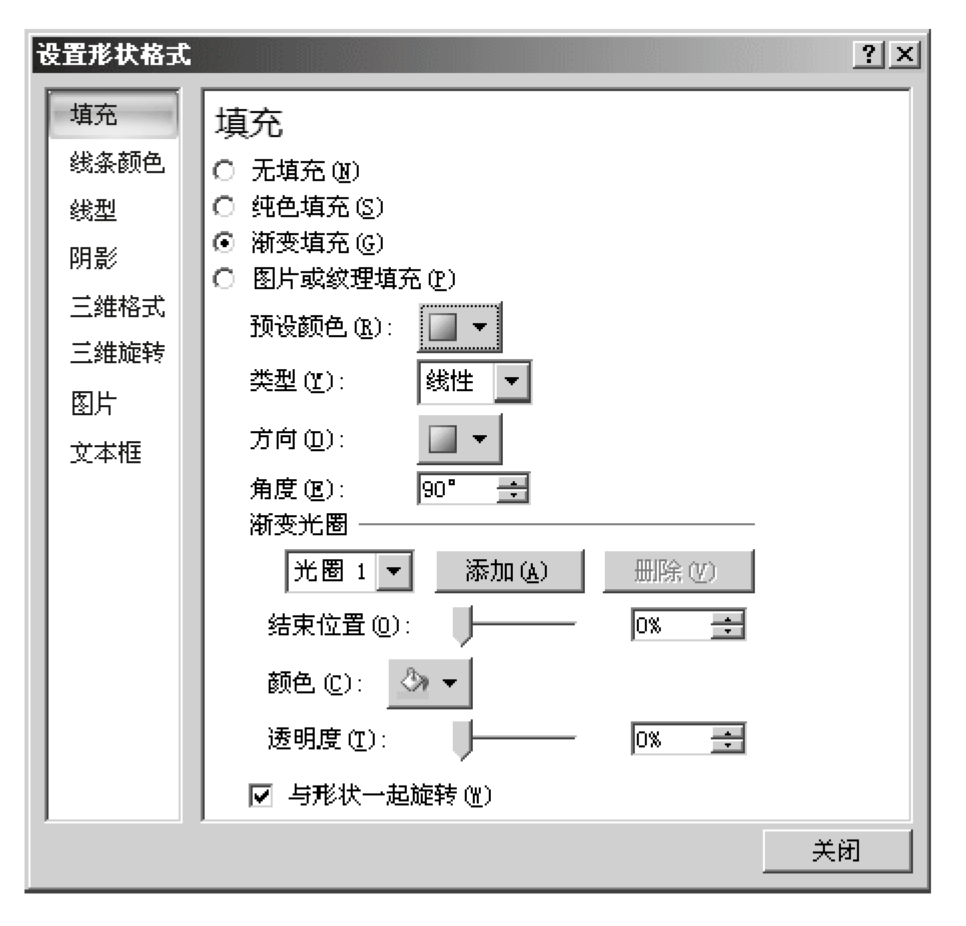


图 11-9　形状填充设置



图 11-10　指定登录系统宏

11.3.2　首页代码设计

首页按钮的宏代码被保存在“首页按钮的宏代码”模块中，打开该模块后即可见到所有宏代码。模块包含了 6 个过程，分别对应 6 个按钮。这些过程的代码大致一致，但有些模块需要检测是否有用户登录系统，只有用户登录后才能开启相应窗口。例如合同基本信息管理、合同收费信息管理、合同信息查询与导出 3 个模块，登录系统、修改用户名和修改密码中都需要输入用户名和密码，因此不需要用户登录：以下是这些宏的详细代码解释：

```
Sub 基本信息()
If IsLogin Then                                                         '检测是否有用户登录系统
    合同基本信息管理.Show                                               '显示合同基本信息管理窗口
Else
    MsgBox "你还没有登录系统！", vbInformation + vbOKOnly               '显示没有用户登录的提示信息
End If
End Sub

Sub 收费信息()
If IsLogin Then                                                         '检测是否有用户登录系统
    合同收费信息管理.Show                                               '显示合同收费信息管理窗口
Else
    MsgBox "你还没有登录系统！", vbInformation + vbOKOnly               '显示没有用户登录的提示信息
End If
End Sub

Sub 信息查询与导出()
If IsLogin Then                                                         '检测是否有用户登录系统
    合同信息查询与导出.Show                                             '显示合同信息查询与导出窗口
Else
    MsgBox "你还没有登录系统！", vbInformation + vbOKOnly               '显示没有用户登录的提示信息
End If
End Sub
```

```
Sub 用户名()
修改用户名.Show                                  '显示修改用户名窗口
End Sub

Sub 密码()
修改密码.Show                                    '显示修改密码窗口
End Sub

Sub 登录系统()
用户登录.Show                                    '显示用户登录窗口
End Sub
```

11.4 模块代码设计

在窗口代码设计中使用到了部分公共变量及公用过程，这些公共变量和过程都被单独保存在模块代码中。为了便于介绍窗口代码设计，这里首先介绍模块代码，以便读者阅读后续窗口代码。

实例包含了 3 个模块，前面首页代码设计部分已经讲述了“首页按钮的宏代码”模块代码。这里将分两节讲述剩余的两个模块：公共变量模块和创建数据库程序模块。

11.4.1 公共变量模块代码设计

公共变量模块中包含了程序运行中使用到的所有公共变量，了解这部分变量的作用有助于理解整个系统的架构和代码设计方式。以下是该模块包含的代码：

```
Public cnn As ADODB.Connection          '公用数据库链接对象
Public rs As ADODB.Recordset            '公用数据记录集对象
Public myArray As Variant               '自定义数组，具体作用在后续代码中详细介绍
Public IsLogin As Boolean               '确认当前是否有用户登录系统
```

11.4.2 创建数据库程序模块代码设计

创建数据库程序模块仅包含了一个过程：创建数据库。该过程用于检测数据库是否存在，当数据库不存在时，程序通过代码建立该数据库以及各个表的结构。本实例是使用 ADO 数据库对象实现的，读者也可以对照前面使用 ADO 数据库对象的建立方法。在实际情况中，读者可以自行选择一种方法。如图 11-11 所示的是该过程的代码执行流程。

该过程的详细代码解释如下所示：

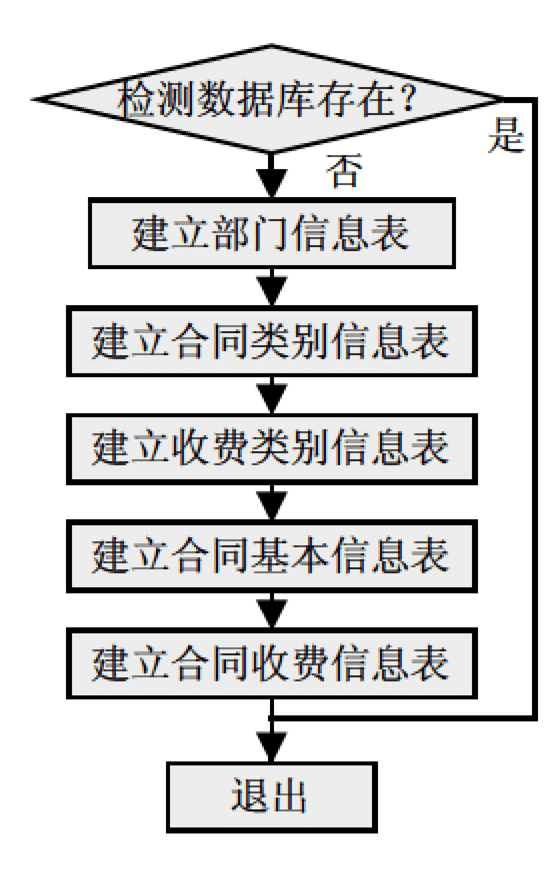


图 11-11 创建数据库过程流程图

```
Public Sub 创建数据库()
    Dim myCat As New ADOX.Catalog
    Dim myCmd As New ADODB.Command
    Dim mydata As String
    Dim SQL As String
    mydata = ThisWorkbook.Path & "\合同管理.mdb"                    '数据库保存位置
    If Dir(mydata) = "" Then                                          '检查数据库是否存在
        MsgBox "数据库<合同管理.mdb>不存在！下面将创建这个数据库！", _
            vbExclamation, "创建数据库"
        '创建数据库文件
        myCat.Create "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & mydata
        '设置数据库连接
        Set myCmd.ActiveConnection = myCat.ActiveConnection
        '创建数据表"部门信息"
        SQL = "CREATE TABLE 部门信息(部门名称 text(20))"
        myCmd.CommandText = SQL
        myCmd.Execute , , adCmdText
        '创建数据表"合同类别信息"
        SQL = "CREATE TABLE 合同类别信息(合同类别 text(20))"
        myCmd.CommandText = SQL
        myCmd.Execute , , adCmdText
        '创建数据表"收费类别信息"
        SQL = "CREATE TABLE 收费类别信息(收费类别 text(20))"
        myCmd.CommandText = SQL
        myCmd.Execute , , adCmdText
        '创建数据表"合同基本信息"
        SQL = "CREATE TABLE 合同基本信息(合同号 text(20),项目名称 text(30)," _
            & "委托单位 text(50),联系人 text(10),联系电话 text(20),签订日期 date," _
            & "签订人 text(10),签订部门 text(20)," _
            & "合同起始日期 date,合同终止日期 date,合同金额 currency," _
            & "合同类别 text(20),备注 text(50))"
        myCmd.CommandText = SQL
        myCmd.Execute , , adCmdText
        '创建数据表"合同收费信息"
        SQL = "CREATE TABLE 合同收费信息(合同号 text(20),收费类别 text(20)," _
            & "收费日期 date,收费金额 currency,备注 text(50))"
        myCmd.CommandText = SQL
        myCmd.Execute , , adCmdText
        '释放变量
        Set myCat = Nothing
        Set myCmd = Nothing
        '弹出信息
        MsgBox "创建数据库成功！" & vbCrLf _
            & "数据库文件名为：合同管理.mdb" & vbCrLf _
            & "保存位置：" & ThisWorkbook.Path, vbInformation, "创建数据库"
    End If
End Sub
```

11.5　用户登录窗口设计

该系统打开后，会立即弹出用户登录窗口。用户可以退出该窗口取消登录，但在首页单击【用户登录】按钮又可重新开启该窗口。只有当用户登录系统后，才可以完成与合同相关的所有操作。但仍然可以修改用户名以及密码，可以在不登录系统时进行。

11.5.1　用户登录窗口界面设计

用户登录窗口界面十分简单，包含的控件数量很少，这里不再以列表形式体现控件列表明细。窗口包含了 2 个标签控件、2 个文本框控件以及 2 个按钮控件。标签以及文本框分为两组，分别用于用户名和用户密码的提示与输入工作。两个按钮分别完成确认登录与关闭窗口工作。如图 11-12 所示是该窗口的界面。

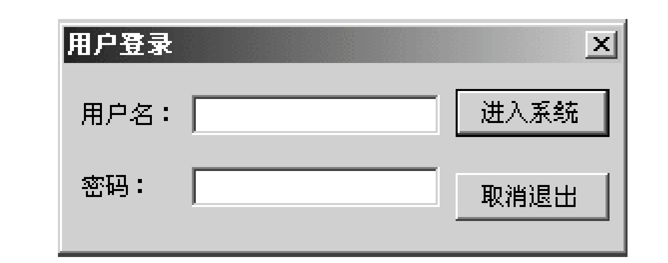


图 11-12　用户登录窗口界面

建立该窗口的步骤如下：

（1）在 VBE 环境中依次选择【插入】|【用户窗体】命令建立一个新窗体。在属性窗口中将该新窗体的名称属性设置为“用户登录”，如图 11-13 所示。

（2）在工具箱中选择标签控件，在窗体中连续插入两个标签。在属性窗口中分别将两标签控件的名称设置为“用户名：”和“密码：”，如图 11-14 所示。



图 11-13　用户登录窗体属性设置

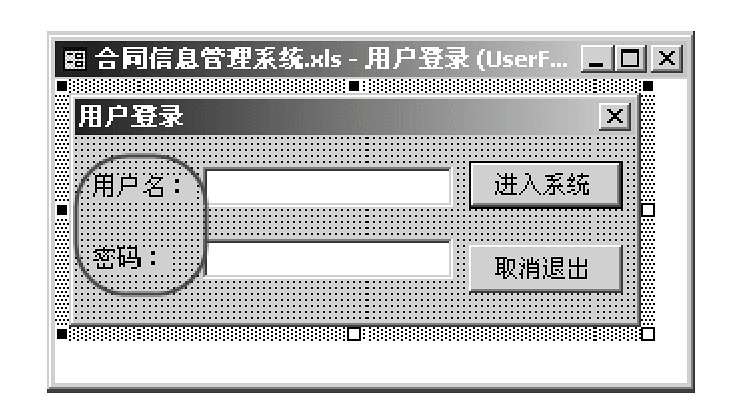


图 11-14　用户登录窗体设计效果

（3）在工具箱中选择文本框控件，在窗体中连续插入两个文本框。在属性窗口中分别将两文本框控件的名称设置为 TextBox1 和 TextBox2。

（4）在工具箱中选择按钮控件，在窗体中连续插入两个按钮控件。在属性窗口中分别将两按钮的名称设置为 CommandButton1 和 CommandButton2，Caption 属性分别设置为“进入系统”和“取消退出”。

11.5.2　窗体代码设计

窗体使用了一个布尔变量 IsLogin，通过该变量程序确认是否有用户已经登录。该变量的状态决定了用户是否可以使用与合同相关的功能。但用户仍然可以进行用户名、用户密码修改工作。因为在这些修改过程中仍然需要用户输入用户名和用户密码。

窗口包含了 3 个过程，分别是窗口初始化事件、确认按钮单击事件、关闭按钮单击事件。在初始化事件中，过程完成窗口中各控件的状态初始化任务。确认按钮单击事件检查用户名是否存在、密码是否正确以及登记用户登录、打开数据库链接等工作。关闭按钮单击事件用于退出系统，关闭工作簿。

在窗口初始化代码中不要设置 isLogin 公共变量的值为 False，因为可能已经有用户登录成功。此时只需要保证 isLogin 的值不变化即可，以防造成混乱。窗口初始化事件代码如下：

```
Private Sub UserForm_Initialize()
    TextBox1 = ""                    '置空用户名文本框
    TextBox2 = ""                    '置空密码文本框
    TextBox1.SetFocus                '将焦点定位到用户名文本框
End Sub
```

确认按钮的单击事件代码比较繁杂。这里将列出该过程的流程图，读者可以参照该图理解该过程的代码。该过程的流程图如图 11-15 所示。

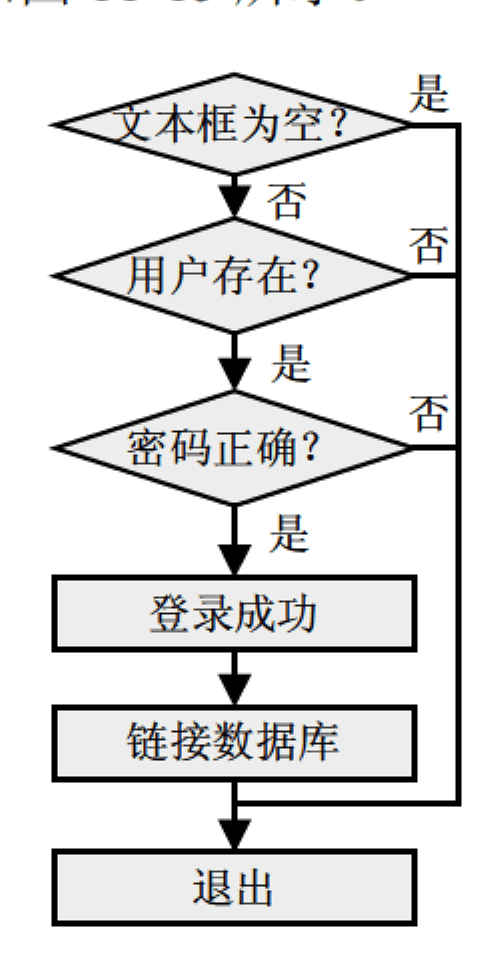


图 11-15　登录系统窗口流程图

```
Private Sub CommandButton1_Click()
    On Error GoTo errorhandle
    Dim ws As Worksheet
    Set ws = Worksheets("用户名密码")                    '获得用户名密码工作表对象
    If TextBox1.Text = "" Then                           '检查用户名文本框是否为空
        TextBox1.SetFocus                                '重置鼠标焦点到用户名文本框
        Exit Sub
    End If
    If TextBox2.Text = "" Then                           '检查密码文本框是否为空
        TextBox2.SetFocus                                '重置鼠标焦点到密码文本框
        Exit Sub
    End If
    '循环对比用户名密码表中的用户名、密码与用户输入用户名、密码
    For i = 2 To ws.Range("A65536").End(xlUp).Row
        If ws.Range("A" & i).Text = TextBox1.Text _
            And ws.Range("B" & i).Text = TextBox2.Text Then     '找到对应用户名与密码
```

```
            Unload 用户登录                                             '退出窗口
            IsLogin = True                                              '设置登录标识为真
            Call 创建数据库                                             '链接数据库或建立数据库
            Exit Sub
        ElseIf ws.Range("A" & i).Text = TextBox1.Text _
            And ws.Range("B" & i).Text <> TextBox2.Text Then            '用户名正确但密码不正确
            MsgBox "密码错误！", vbCritical, "警告"                      '提示密码错误
            TextBox2 = ""                                               '置空密码文本框
            TextBox2.SetFocus                                           '将鼠标焦点等位到密码文本框
            Exit Sub
        ElseIf ws.Range("A" & i).Text <> TextBox1.Text _
            And ws.Range("B" & i).Text = TextBox2.Text Then             '用户名不正确，密码正确
            MsgBox "用户名错误！", vbCritical, "警告"                    '提示用户名错误
            TextBox1 = ""                                               '置空用户名文本框
            TextBox1.SetFocus                                           '将鼠标焦点定位到用户名文本框
            Exit Sub
        End If
    Next i
    MsgBox "用户名和密码不存在！", vbCritical, "警告"                    '提示用户名和密码错误
    TextBox1 = ""
    TextBox2 = ""
    TextBox1.SetFocus
    Unload 用户登录
errorhandle:
    ThisWorkbook.Close savechanges:=False                               '关闭工作簿并且不保存
End Sub
```

单击【关闭】按钮时，完成的工作比较简单，只需要退出系统，并且不保存工作簿的变化。以下是该按钮单击事件的代码：

```
Private Sub CommandButton2_Click()
    ThisWorkbook.Close savechanges:=False
End Sub
```

11.6 修改用户名窗口设计

修改用户名窗口用于修改用户的用户名，该窗口可以直接被打开，无须登录系统。在修改用户名过程中，需要输入正确的新用户名、原用户名和密码方可修改成功。在该窗口中修改用户名将不会造成该用户登录系统，并且当前用户修改用户名时仍然需要输入密码。

11.6.1 窗口界面设计

该窗体包含了比较少的控件，此处只做简要的文字介绍，不再用表列明控件。窗体共包含了 3 个标签控件、3 个文本框控件以及 2 个按钮控件。1 个标签控件对应 1 个文本框控件。【确定】按钮用于确定用户名修改，【取消】按钮用于关闭窗口。该窗口的界面如图 11-16 所

示。

创建该窗口的步骤如下：

（1）在 VBE 环境中依次选择【插入】|【用户窗体】命令建立一个新窗体。在属性窗口中将该新窗体的名称属性设置为“修改用户名”，如图 11-17 所示。



图 11-16　修改用户名窗口界面



图 11-17　修改用户名窗体属性设置

（2）在工具箱中选择标签控件，在窗体中连续插入 3 个标签。在属性窗口中分别将 3 个标签控件的 Caption 属性设置为“原用户名：”、“新用户名：”和“密码：”，如图 11-18 所示。



图 11-18　修改用户名窗体设计效果

（3）在工具箱中选择文本框控件，在窗体中连续插入 3 个文本框。在属性窗口中分别将 3 个文本框控件的名称属性设置为 TextBox1，TextBox2 和 TextBox3。

（4）在工具箱中选择按钮控件，在窗体中连续插入 3 个按钮控件。在属性窗口中分别将 3 个按钮的名称设置为 CommandButton1、CommandButton2 和 CommandButton3，Caption 属性设置为“确定”和“取消”。

11.6.2　窗口代码设计

窗口包含的事件代码不多，只有两个按钮的单击事件代码。而且大部分代码集中到【确定】按钮中。单击【确定】按钮时，首先检测该用户是否存在，然后确认密码是否正确。当用户名和密码同时正确时，程序将在用户名密码表中对该用户的用户名做出修改并保存。图 11-19 列出了该按钮单击事件过程的流程图。

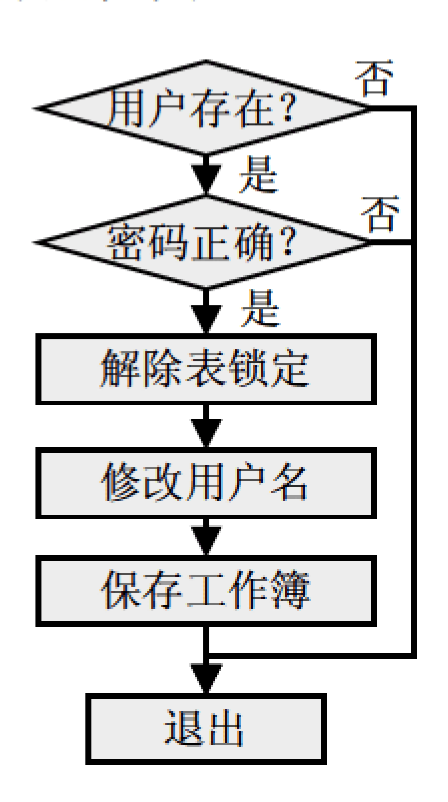


图 11-19　修改用户名过程流程图

```
Private Sub CommandButton1_Click()
    On Error GoTo errorhandle
    Dim ws As Worksheet, noExist As Boolean
    noExist = False                                               '初始化 noExist 变量
    Set ws = Worksheets("用户名密码")                              '定义用户名密码表对象
    For i = 2 To ws.Range("A65536").End(xlUp).Row                 '循环检测用户名和密码
        If ws.Range("A" & i).Text = TextBox1.Text Then            '检测用户名是否正确
            If ws.Range("B" & i).Text = TextBox3.Text Then        '检测密码是否正确
                ws.Unprotect Password:="123"                      '解锁工作表
                ws.Range("A" & i) = TextBox2.Text                 '修改用户名
                ws.Range("C" & i) = Now()                         '登记用户名修改时间
                ws.Protect Password:="123"                        '锁定工作表
                TextBox1.Text = ""                                '重置原用户名文本框
                TextBox2.Text = ""                                '重置新用户名文本框
                TextBox3.Text = ""                                '重置密码文本框
                MsgBox "用户名修改成功！请记好您的新用户名！", _          '提示用户名修改成功
                    vbInformation, "用户名修改成功"
                Unload 修改用户名
                '保存工作簿
                ThisWorkbook.Save
                Exit Sub                                          '退出过程
            Else
                MsgBox "输入密码错误！", vbInformation + vbOKOnly       '提示密码错误
                TextBox3.Text = ""                                '重置密码文本框
                TextBox3.SetFocus                                 '将鼠标焦点定位到密码文本框
            End If
        Else
            noExist = True                                        '标识未找到该用户名
        End If
    Next i
    If noExist = True Then
        MsgBox "没有用户名 " & TextBox1.Text & " ！", vbCritical, "警告"     '提示未找到用户时
        Unload 修改用户名                                          '退出窗口
    End If
errorhandle:
    If Err.Number <> 0 Then
        MsgBox "错误！" + CStr(Err.Description), vbCritical, "错误"        '提示错误信息
    End If
End Sub
```

【取消】按钮的代码十分简单，仅仅退出窗口即可。下面代码是该按钮的单击事件代码：

```
Private Sub CommandButton2_Click()
    Unload 修改用户名
End Sub
```

11.7 修改密码窗口设计

修改密码窗口和修改用户名窗口的结构大致类似。该窗口也可以被直接打开，无须登录系统。在修改密码过程中，需要输入用户名、原密码以及新密码。修改密码也不会造成该用户登录系统，用户只有在用户登录窗口中完成登录方被确认登录系统。

11.7.1 修改密码窗口界面设计

修改密码窗口的界面和修改用户名的窗口界面类似。唯一的不同之处在于本窗口修改的对象是用户密码。该窗口的界面效果图如图 11-20 所示。该窗口包含了 3 个标签控件、3 个文本框控件以及 2 个按钮控件。1 个标签控件和 1 个文本框控件相互对应。



图 11-20 修改密码窗口界面

创建该窗口的步骤如下：

（1）在 VBE 环境中依次选择【插入】|【用户窗体】命令建立一个新窗体。在属性窗口中将该新窗体的名称属性设置为“修改密码”，如图 11-21 所示。

（2）在工具箱中选择标签控件，在窗体中连续插入 3 个标签。在属性窗口中分别将 3 个标签控件的 Caption 属性设置为“用户名：”、“原密码：”和“新密码：”，如图 11-22 所示。



图 11-21 修改密码窗体属性设置

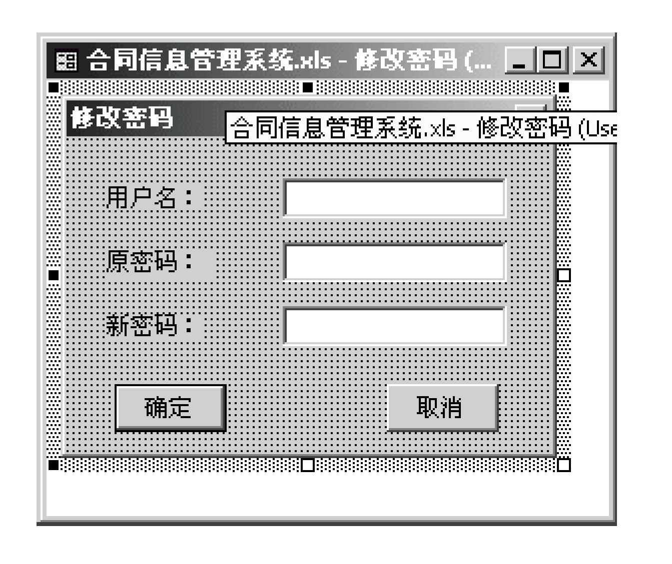


图 11-22 修改密码窗体设计效果

（3）在工具箱中选择文本框控件，在窗体中连续插入 3 个文本框。在属性窗口中分别将 3 个文本框控件的名称设置为 TextBox1，TextBox2 和 TextBox3。

（4）在工具箱中选择按钮控件，在窗体中连续插入 3 个按钮控件。在属性窗口中分别将 3 个按钮的名称设置为 CommandButton1、CommandButton2 和 CommandButton3，Caption 属性设置为“确定”和“取消”。

11.7.2 窗口代码设计

窗口代码包含了 2 个过程，分别是【确认】按钮与【取消】按钮的单击事件过程。【确认】按钮被单击时，首先会对密码进行检测，确认密码位数至少为 5 位数，然后在用户名密

码表中查找该用户名及对应的密码。当找到该用户名且密码正确时，程序将修改该用户的密码为新密码。

```
Private Sub CommandButton1_Click()
    On Error GoTo errorhandle
    Dim ws As Worksheet, noExist As Boolean
    noExist = False
    Set ws = Worksheets("用户名密码")
    If Len(TextBox2.Text) < 5 Then
        MsgBox "密码位数最少不能小于 5 位！ ", vbCritical, "注意"
        TextBox2.Text = ""
        TextBox3.Text = ""
        TextBox2.SetFocus
        Exit Sub
    End If
    For i = 2 To ws.Range("A65536").End(xlUp).Row               '循环检测用户名和密码
        If ws.Range("A" & i).Text = TextBox1.Text Then          '检测用户名是否正确
            If ws.Range("B" & i).Text = TextBox2.Text Then      '检测密码是否正确
                ws.Unprotect Password:="123"                    '解锁工作表
                ws.Range("A" & i) = TextBox3.Text               '修改密码
                ws.Range("C" & i) = Now()                       '登记用户名修改时间
                ws.Protect Password:="123"                      '锁定工作表
                TextBox1.Text = ""                              '重置原用户名文本框
                TextBox2.Text = ""                              '重置新用户名文本框
                TextBox3.Text = ""                              '重置密码文本框
                MsgBox "密码修改成功！请记好您的新密码！ ", _     '提示密码修改成功
                    vbInformation, "用户名修改成功"
                Unload 修改用户名
                '保存工作簿
                ThisWorkbook.Save
                Exit Sub                                            '退出过程
            Else
                MsgBox "输入密码错误！ ", vbInformation + vbOKOnly      '提示密码错误
                TextBox2.Text = ""                          '重置密码文本框
                TextBox2.SetFocus                           '将鼠标焦点定位到密码文本框
            End If
        Else
            noExist = True                                  '标识未找到该用户名
        End If
    Next i
    If noExist = True Then
        MsgBox "没有用户名 " & TextBox1.Text & " ！ ", vbCritical, "警告"
        Unload 修改密码
    End If
errorhandle:
    If Err.Number <> 0 Then
        MsgBox "错误！ " + CStr(Err.Description), vbCritical, "错误"
    End If
End Sub
```

代码说明：

- 程序中对密码的长度进行了限制，必须保证密码长度在 5 位以上。当密码的长度不能达到要求时，程序将把密码输入框置空，要求重输。
- 循环检测用户名和密码时使用了 If 语句的嵌套结构。通过该结构的双重判断区分用户名输入错误和密码输入错误两种情况。

退出该窗口时，代码和其他窗口的退出代码一致，仅需要使用 Unload 命令完成窗口的卸载工作即可。退出窗口按钮的代码如下：

```
Private Sub CommandButton2_Click()
    Unload 修改密码
End Sub
```

11.8　合同基本信息管理窗口设计

在合同基本信息管理窗口中可以完成对合同基本信息资料的浏览、添加、修改和删除等基本操作。从本节开始到本章结束，大量使用 ADO 数据库对象操作数据库文件的表数据，读者由此可以熟悉 ADO 数据库对象操作数据库时的方法。

11.8.1　窗口界面设计

该窗口控件数量繁多，本小节将详细讲述该窗口包含的控件和建立步骤。本窗口的界面如图 11-23 所示。



图 11-23　合同基本信息管理窗口界面

该窗体一共包含了 2 个框架控件、19 个标签控件、13 个文本框控件、2 个复合框控件、12 按钮控件和 1 个 ListView 控件。表 11-5 列出了控件的名称、类型、功能和属性设置说明。限于篇幅，列表不再对标签控件做介绍，该控件只需要修改对应的 Caption 属性即可。

表 11-5　合同基本信息管理窗口控件列表

控　件　名	控 件 类 型	控 件 说 明
Frame1	框架控件	包含了建立合同时所有的基本信息输入项目，该框架将这部分同类控件与其他控件区分开来。该控件的 Caption 属性被设置为“合同基本信息”
Frame2	框架控件	包含了 ListView 控件，该部分用于显示对应合同的所有收费信息。该控件的 Caption 属性被设置为“合同收费情况”
合同号	文本框	该文本框用于显示或设置合同号
合同类别	复合框	该复合框用于显示或设置合同的类别
添加类别	按钮	该按钮用于添加新合同类别。在合同类别复合框中输入新合同类别后，选择添加类别即可
项目名称	文本框	该文本框用于显示或设置项目名称
委托单位	文本框	该文本框用于显示或设置委托单位
联系人	文本框	该文本框用于显示或设置联系人
联系电话	文本框	该文本框用于显示或设置联系电话
签订人	文本框	该文本框用于显示或设置签订人
签订部门	复合框	该复合框用于显示或设置签订部门
添加部门	按钮	该按钮用于添加新部门。在部门复合框中输入新部门后，选择添加部门即可
签订日期	文本框	该文本框用于显示或设置签订日期
合同起始日	文本框	该文本框用于显示或设置合同起始日期
合同终止日	文本框	该文本框用于显示或设置合同终止日期
合同金额	文本框	该文本框用于显示或设置合同金额
收费合计	文本框	该文本框用于显示或设置合同收费合计
欠费合计	文本框	该文本框用于显示或设置合同欠费合计
备注	文本框	该文本框用于显示或设置合同备注内容
ListView1	ListView	显示当前窗体上所显示合同的详细收费情况
新合同	按钮	该按钮用于重置所有文本框和复合框，以便输入新合同。ListView 控件的项目也将被清空
添加	按钮	该按钮依据在窗口中输入的内容在数据库中建立新合同
修改	按钮	该按钮将对显示在窗口中的当前合同基本信息进行修改
删除	按钮	该按钮将删除显示在窗口中的当前合同基本信息
第一条	按钮	将当前显示的合同基本信息定位到第一条
下一条	按钮	将当前显示的合同基本信息定位到下一条
上一条	按钮	将当前显示的合同基本信息定位到上一条
最末条	按钮	将当前显示的合同基本信息定位到最后一条
查询	按钮	显示符合当前合同号的合同基本信息
退出	按钮	退出合同基本信息管理窗口

创建该窗口的步骤如下：

（1）在 VBE 环境中依次选择【插入】|【用户窗体】命令建立一个新窗体。并在属性窗口中设置该窗体的名称属性为“合同基本信息管理”，如图 11-24 所示。

（2）在工具箱中选择框架控件。在窗体上插入一个框架，在属性窗口中修改该框架的 Caption 属性为“合同基本信息”，名称修改为 Frame1，如图 11-25 所示。



图 11-24　合同基本信息管理窗体属性设置

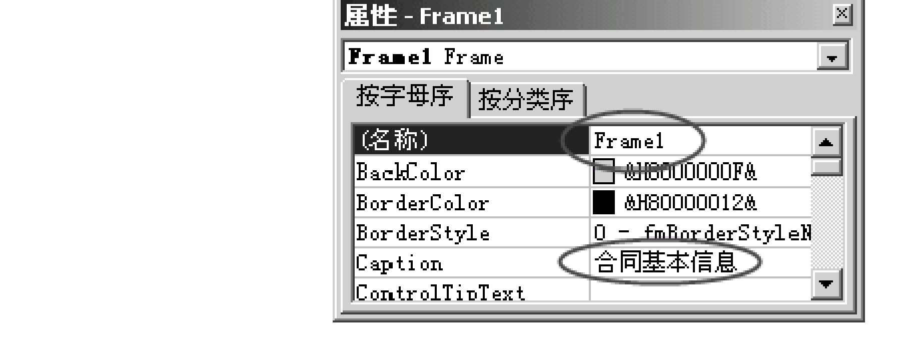


图 11-25　合同基本信息框架控件属性设置

（3）在工具箱中选择标签控件，依次在合同基本信息框架中添加 15 个标签控件。然后在属性窗口中分别设置这些控件的 Caption 属性为“合同号”、“合同类别”、“项目名称”、“委托单位”、“联系人”、“联系电话”、“签订人”、“签订部门”、“签订日期”、“合同起始日”、“合同终止日”、“合同金额”、“收费合计”、“欠费合计”和“备注”，如图 11-26 所示。

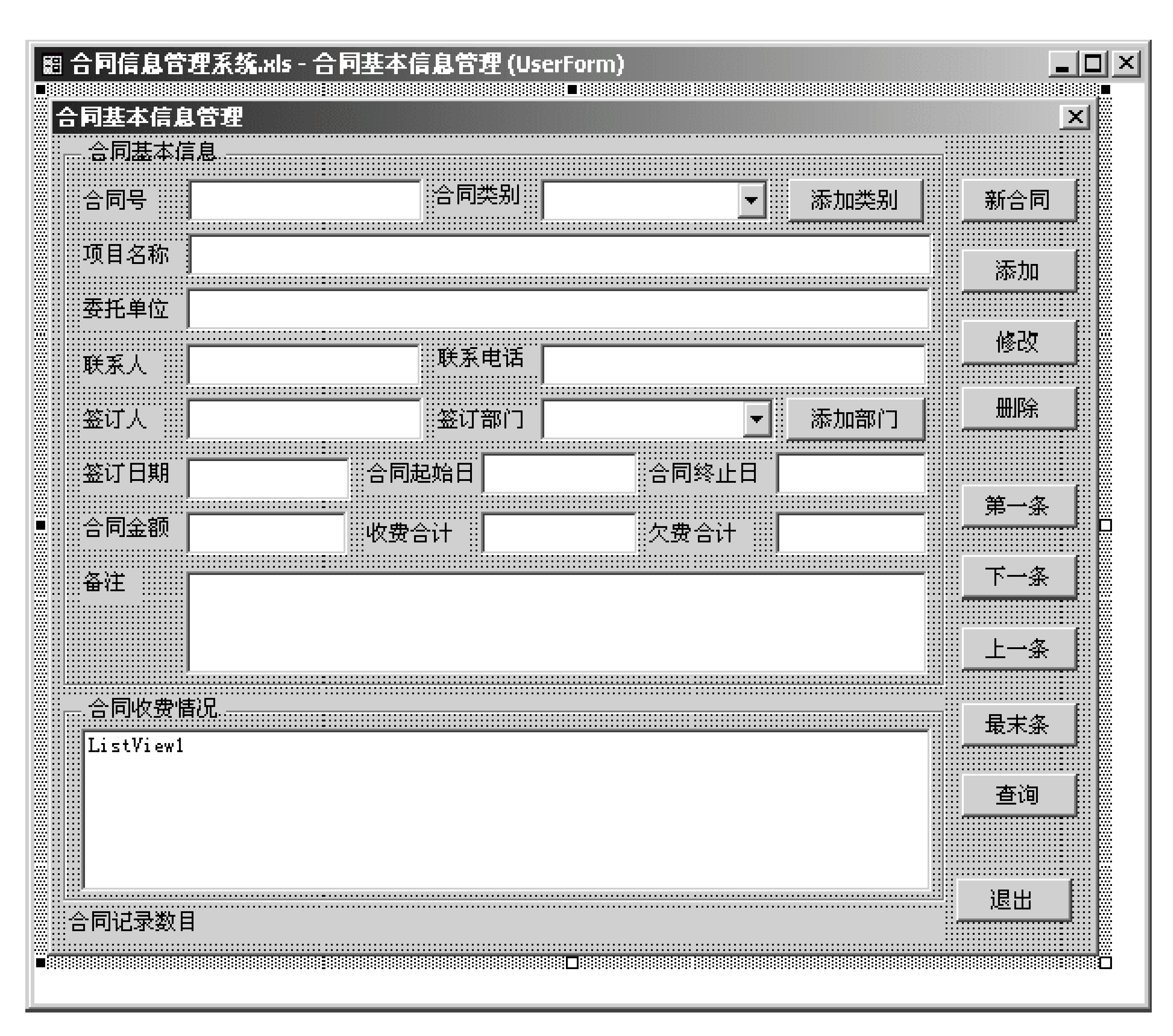


图 11-26　合同基本信息管理窗体设计效果

（4）在工具箱中选择文本框控件，依次在合同基本信息框架中添加 13 个文本框控件。然后在属性窗口中分别设置这些控件的名称属性为“合同号”、“项目名称”、“委托单位”、“联系人”、“联系电话”、“签订人”、“签订日期”、“合同起始日”、“合同终止日”、“合同金额”、“收费合计”、“欠费合计”和“备注”，SelectionMargin 属性都设置为 False，如图 11-27 所示。

（5）在工具箱中选择复合框控件，依次在合同基本信息框架中添加 2 个复合框控件。随后在属性窗口中分别设置这些控件的名称属性为：“合同类别”、“签订部门”，SelectionMargin 属性都设置为 False。

（6）在工具箱中选择按钮控件，依次在合同基本信息框架中添加 2 个按钮控件。随后在属性窗口中分别设置两个按钮控件的名称属性为“添加类别”、“添加部门”。

（7）调整合同基本信息框架中所有控件的大小与位置，使这些控件的大小与位置与如图 11-27 所示窗口相似。

（8）在工具箱中选择框架控件。在窗体上插入一个框架。随后在属性窗口中设置该框架的 Caption 属性为“合同收费情况”，名称修改为 Frame2，如图 11-28 所示。



图 11-27　设置文本框的 SelectionMargin 属性

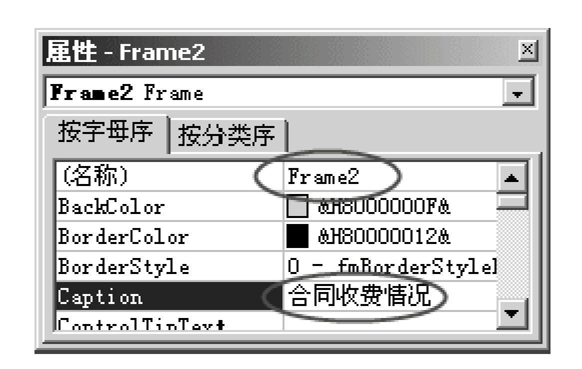


图 11-28　合同收费情况框架属性设置

（9）选择 ListView 控件，在合同收费情况框架中插入一个 ListView 控件。随后在属性窗口中设置名称属性设置为 ListView1。

（10）在合同收费情况框架的下方插入一标签控件。随后在属性窗口中设置该标签控件的 Caption 属性为“合同记录数目”。其他属性默认即可。

（11）在工具箱中选择按钮控件。在窗体上依次插入 10 个按钮。随后在属性窗口中设置这些按钮的 Caption 属性过依次为“新合同”、“添加”、“修改”、“删除”、“第一条”、“下一条”、“上一条”、“最末条”、“查询”和“退出”。

11.8.2　窗口初始化与关闭事件代码设计

本窗体的代码较长，下面将分不同类别对该窗体的代码加以介绍。本小节讲述的是该窗体的初始化与关闭事件代码。窗口初始化过程完成的工作主要是设置窗体控件状态、链接数据库、设置复合框、显示合同基本信息、显示合同收费信息。如图 11-29 显示的是该窗体初始化过程的流程图。

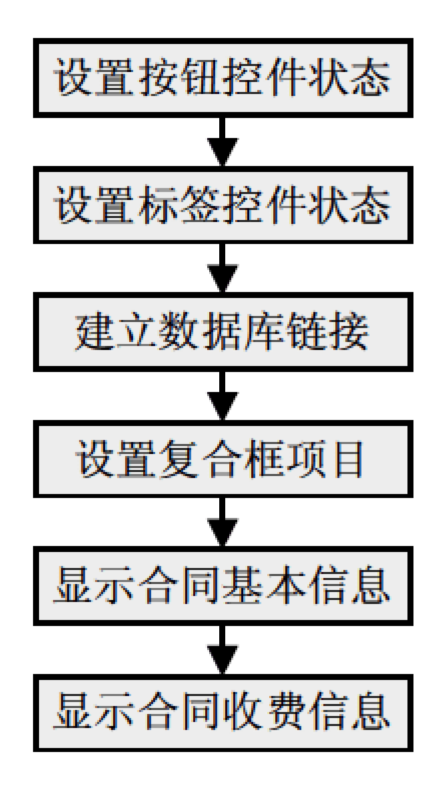


图 11-29　合同基本信息管理窗口初始化过程流程图

```
Private Sub UserForm_Initialize()
    Dim mydata As String
    Dim i As Integer
    '指定数据库
```

```
    mydata = ThisWorkbook.Path & "\合同管理.mdb"
    '设置窗体控件组（也是数据表的各个字段组）
    myArray = Array("合同号", "项目名称", "委托单位", "联系人", "联系电话", _
        "签订日期", "签订人", "签订部门", "合同起始日", "合同终止日", _
        "合同金额", "合同类别", "备注")
    '设置窗体控件的背景颜色
    For i = 0 To UBound(myArray)
        Me.Controls(myArray(i)).BackColor = &HE0E0E0
    Next
    '设置收费合计和欠费合计两个文本框及其对应标签的背景颜色和可操作性
    收费合计.BackColor = &HE0E0E0
    欠费合计.BackColor = &HE0E0E0
    收费合计.Enabled = False
    欠费合计.Enabled = False
    收费合计标签.Enabled = False
    欠费合计标签.Enabled = False
    '设置合同记录数目标签的背景颜色和前景颜色以及显示效果
    合同记录数目.SpecialEffect = fmSpecialEffectSunken
    合同记录数目.ForeColor = &HC000C0
    合同记录数目.BackColor = &HC0C0C0
    '建立与数据库的连接
    Set cnn = New ADODB.Connection
    With cnn
        .Provider = "microsoft.jet.oledb.4.0"
        .Open mydata
    End With
    '调用子程序，为合同类别复合框设置项目
    Call 合同类别复合框设置
    '调用子程序，为签订部门复合框设置项目
    Call 签订部门复合框设置
    '调用子程序,查询合同基本信息
    Call 查询合同基本信息
    '调用子程序,显示合同基本信息
    Call 显示合同基本信息
    '调用子程序,显示某合同的收费信息
    Call 显示合同收费情况
End Sub
```

代码说明：

- 在窗口初始化过程中，包含了对 3 类控件的状态设置。这些控件包括文本框和复合框、按钮、标签。在设置文本框和复合框时，使用了一个数组保存了这些文本框和复合框的名称，然后使用一个 For 循环访问这些控件，并修改相应的 BackColor 属性。访问这些控件时，使用窗口的 Controls 集合。当控件数量众多，且设置的属性大致相当时，可以采用这样的方式精简代码。
- 在初始化过程中，涉及到的自定义过程将在后续相应小节单独加以介绍。这些过程包括合同类别复合框设置、签订部门复合框设置、查询合同基本信息、显示合同基本信息、显示合同收费情况 5 个自定义过程。

用户单击【退出】按钮时，将会关闭本窗体。【退出】按钮代码十分简单，以下是该按钮的代码。

```
Private Sub 退出_Click()
    cnn.Close
    Set rs = Nothing
    Set cnn = Nothing
    Unload 合同基本信息管理
End Sub
```

11.8.3 复合框设置过程代码设计

在合同基本信息框架中包含了两个复合框：窗口合同类别和签订部门复合框。初始化代码中这两个复合框的设置过程分别用于初始化合同类别、签订部门复合框项目。首先从数据库中获取合同类别、签订部门记录集，然后将这些合同类别、签订部门记录数据依次写入合同类别复合框中。以下是这两个自定义过程的详细代码：

```
Public Sub 合同类别复合框设置()
    Dim rsx As New ADODB.Recordset
    Dim i As Integer
    '获取合同类别记录集
    rsx.Open "合同类别信息", cnn, adOpenKeyset, adLockOptimistic
    With 合同类别
        .Clear                              '清空合同类别复合框
        For i = 1 To rsx.RecordCount
            .AddItem rsx.Fields(0)          '将合同类别记录数据写入合同类别复合框中
            rsx.MoveNext                    '将当前记录移到下一条
        Next
    End With
    rsx.Close                               '关闭记录集
    Set rsx = Nothing                       '清空记录集对象占用内存空间
End Sub

Public Sub 签订部门复合框设置()
    Dim rsx As New ADODB.Recordset
    Dim i As Integer
    '获取部门信息记录集
    rsx.Open "部门信息", cnn, adOpenKeyset, adLockOptimistic
    With 签订部门
        .Clear                              '清空签订部门复合框
        For i = 1 To rsx.RecordCount
            .AddItem rsx.Fields(0)          '将签订部门记录数据写入签订部门复合框中
            rsx.MoveNext                    '将当前记录移到下一条
        Next i
    End With
    rsx.Close                               '关闭记录集
    Set rsx = Nothing                       '清空记录集对象占用的内存空间
End Sub
```

11.8.4　查询、显示合同基本信息过程代码设计

窗口初始化时，需要在合同基本信息框架中显示一合同的基本信息。该任务由查询、显示合同基本信息两个过程共同完成。查询合同基本信息过程从数据库中获取合同基本信息记录集，将该记录集保存在 rs 公共变量中。显示合同基本信息过程首先检测 rs 记录集是否为空，当为空时，程序将新建一个合同，否则将把该合同的基本信息写入合同基本信息框架中的控件。显示合同基本信息过程稍微显得复杂，如图 11-30 所示的是该过程的流程图。

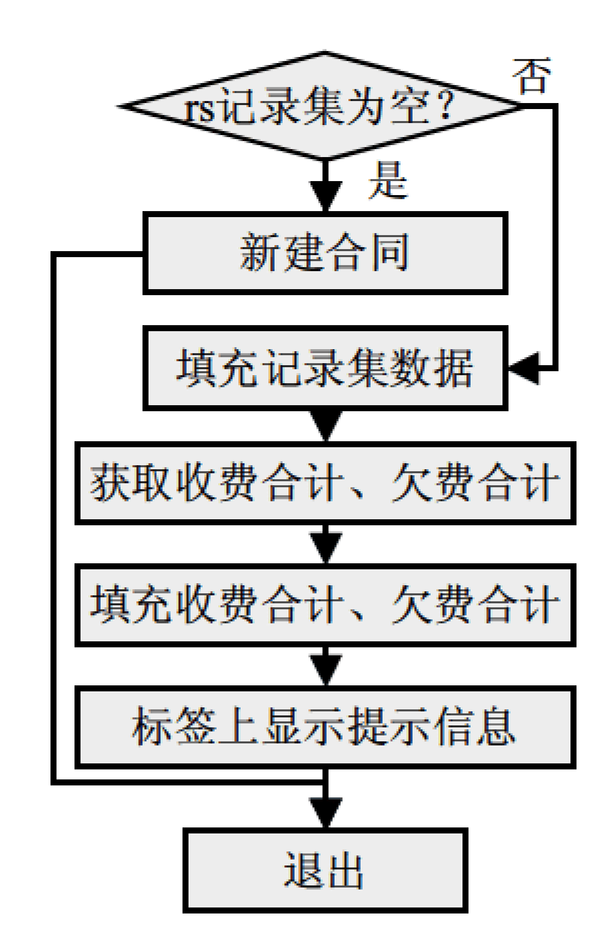


图 11-30　显示合同基本信息过程流程图

以下是这两个过程的详细代码：

```
Public Sub 查询合同基本信息()
    Set rs = New ADODB.Recordset
    '获取合同基本信息记录集
    rs.Open "合同基本信息", cnn, adOpenKeyset, adLockOptimistic
End Sub

Public Sub 显示合同基本信息()
    On Error Resume Next
    Dim i As Integer
    '检测 rs 记录集是否为空
    If rs.BOF And rs.EOF Then
        Call 新合同_Click                                    '新建合同
    Else
        '向合同基本信息框架的控件填充记录集中的数据
        For i = 0 To UBound(myArray)
            If IsNull(rs.Fields(i)) Then
                Me.Controls(myArray(i)).Value = ""
            Else
                Me.Controls(myArray(i)).Value = rs.Fields(i)
            End If
        Next
```

```
        '查询该合同的收费合计和欠费合计
        Dim rsx As New ADODB.Recordset
        SQL = "select sum(收费金额) as aa from 合同收费信息 " _
            & " where 合同号='" & 合同号.Value & "'"
        rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic
        '在窗体上显示该合同的收费合计数据和欠费合计数据
        If IsNull(rsx!aa) Then
            收费合计.Value = 0
        Else
            收费合计.Value = rsx!aa
        End If
        If IsNull(rs.Fields("合同金额")) Then
            欠费合计.Value = 0
        Else
            欠费合计.Value = rs.Fields("合同金额") -Val(收费合计.Value)
        End If
        '在窗体下面的标签中显示数据库中的合同记录总数，以及当前正在显示第几条记录
        合同记录数目.Caption = "数据库中共有 " & rs.RecordCount & " 条合同记录" _
            & Space(5) & "目前是第 " & rs.AbsolutePosition & " 条合同记录"
        rsx.Close
        Set rsx = Nothing
    End If
End Sub
```

11.8.5 显示合同收费情况过程代码设计

合同收费情况框架中的 ListView 控件显示了所有当前合同的收费情况，这些收费情况的数据是从数据库中获得的。显示合同收费情况过程完成的即为该部分任务。该过程首先从数据库中获取一记录集，此记录集的合同号为当前显示合同的合同号，然后过程对 ListView 控件做了显示前的必要设置，最后将记录集中的数据写入到 ListView 控件中。如图 11-31 所示的是该过程的流程图。

获取满足指定条件的记录集 → 设置ListView控件 → 向ListView控件写入数据

图 11-31 显示合同收费情况过程流程图

以下是该过程的详细代码：

```
Public Sub 显示合同收费情况()
    On Error Resume Next
    Dim i As Integer
    Dim SQL As String
    Dim rsx As New ADODB.Recordset
    '获取记录集，该记录集的合同号为当前显示合同的合同号
    SQL = "select * from 合同收费信息 where 合同号='" & 合同号.Value & "'"
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    With ListView1
        '设置 ListView1 的标题、显示类型、整行选择和网格线属性
        .ColumnHeaders.Clear                                    '清除标题
        .ListItems.Clear                                        '清除显示数据
```

```
            .View = lvwReport                                   '设置控件显示类型
            .FullRowSelect = True                               '允许整行选择
            .Gridlines = True                                   '网格线
            '为 ListView1 设置标题
            For i = 0 To rsx.Fields.Count -1
                .ColumnHeaders.Add , , rsx.Fields(i).Name       '添加标题
            Next
            '为 ListView1 设置各行数据
            For i = 1 To rsx.RecordCount
                .ListItems.Add , , rsx.Fields(0).Value          '为 ListView 控件增加新显示项
                For j = 1 To rsx.Fields.Count -1
                    .ListItems(i).SubItems(j) = rsx.Fields(j).Value   '为新增加项添加子项
                Next
                rsx.MoveNext                                    '将记录集指针移到下一条
            Next
        End With
        rsx.Close
        Set rsx = Nothing
End Sub
```

11.8.6　添加类别与部门按钮代码设计

合同基本信息框架包含了合同类别和签订部门两个复合框控件。复合框在窗口初始化时被设置了项目值。用户还可以通过该窗口创建新合同类别和新签订部门。【添加类别】按钮与【添加部门】按钮分别实现各自的添加功能。两按钮的代码十分相似，这里将这两个按钮的单击事件代码设计放置在一起讲述。下面以添加类别为例，添加部门按钮的过程见括号提示。

单击【添加类别】按钮（【添加部门】按钮）后，程序检测新合同类别（新部门）输入是否为空。当不为空时，程序才继续执行，否则退出添加过程。然后程序检测合同类别信息表（部门信息表）是否已经包含当前新类别（新部门），当不包含新类别（新部门）时，程序将该新类别（新部门）写入合同类别信息表（部门信息表）中。所有的写入操作完成后，程序重新设置合同类别（签订部门）复合框项目值，并将复合框的显示值设置为新类别（新部门）。图 11-32 显示的是【添加类别】按钮单击事件过程流程图。

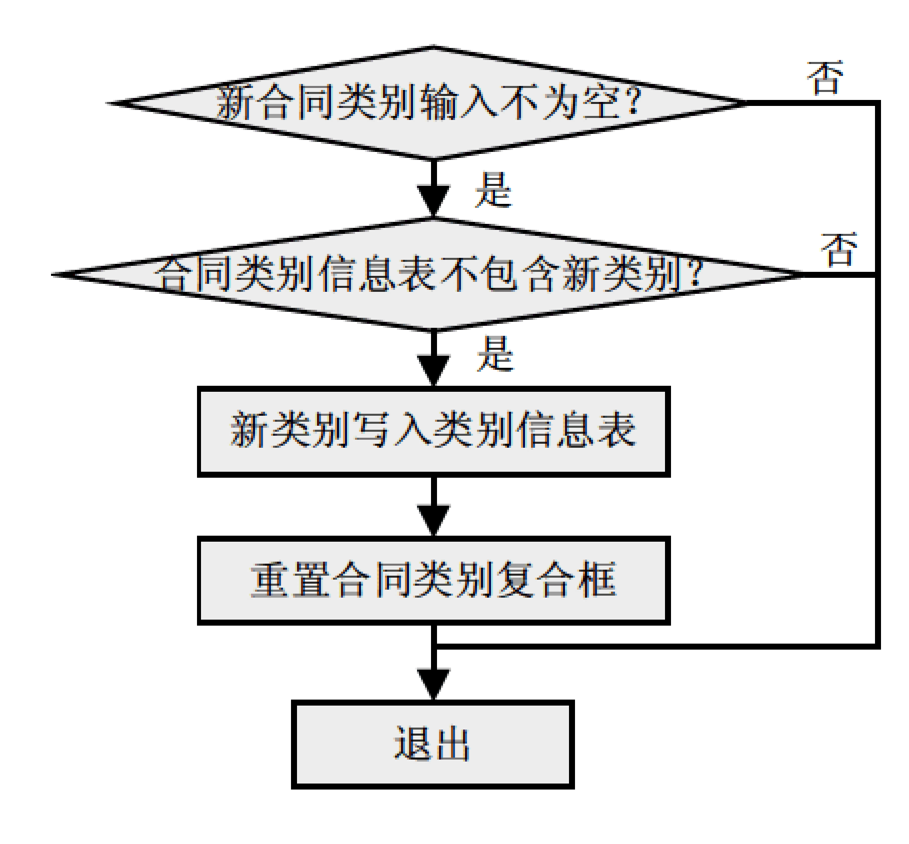


图 11-32　【添加类别】按钮单击事件过程流程图

【添加部门】按钮的单击事件过程流程图可以参照图 11-32，只需要将合同类别换为部门即可。以下是两按钮的详细单击事件代码：

```
Private Sub 添加类别_Click()
    Dim lb As String
    lb = 合同类别.Value                                               '获取新输入的合同类别
    If Len(Trim(lb)) = 0 Then
        MsgBox "没有输入合同类别名称！不能添加！", vbCritical, "警告"      '提示新合同类别为空
        Exit Sub
    End If
    Dim rsx As New ADODB.Recordset
    Dim SQL As String
    SQL = "select * from 合同类别信息 where 合同类别='" & lb & "'"
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic    '获取合同类别和新类别相等的记录集
    If rsx.BOF And rsx.EOF Then                          '当记录集为空时，添加新类别
        rsx.AddNew                                       '记录集添加新记录
        rsx.Fields(0) = lb                               '为新记录赋值
        rsx.Update                                       '更新记录集
        MsgBox "添加完毕！", vbInformation, "添加合同类别"          '提示添加成功
    Else
        MsgBox "已经存在了同名的合同类别名称！", vbCritical, "警告"   '提示新合同类别已经存在
    End If
    rsx.Close                                            '关闭记录集
    Set rsx = Nothing                                    '清除记录集对象占用内存
    Call 合同类别复合框设置                                 '重置合同类别复合框
    合同类别.Value = lb                                    '设置合同类别复合框显示值
End Sub

Private Sub 添加部门_Click()
    Dim bm As String
    bm = 签订部门.Value                                    '获取新输入的签订部门
    If Len(Trim(bm)) = 0 Then
        MsgBox "没有输入签订部门名称！不能添加！", vbCritical, "警告"      '提示新签订部门为空
        Exit Sub
    End If
    Dim rsx As New ADODB.Recordset
    Dim SQL As String
    SQL = "select * from 部门信息 where 部门名称='" & bm & "'"
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic    '获取部门名称与新签订部门相等的记
                                                         录集
    If rsx.BOF And rsx.EOF Then                          '当记录集为空时，添加新部门
        rsx.AddNew                                       '记录集添加新记录
        rsx.Fields(0) = bm                               '为新记录赋值
        rsx.Update                                       '更新记录集
        MsgBox "添加完毕！", vbInformation, "添加部门"      '提示添加部门成功
    Else
        MsgBox "已经存在了同名的部门！", vbCritical, "警告" '提示新部门已经存在
    End If
    rsx.Close                                            '关闭记录集
    Set rsx = Nothing                                    '清除记录集对象占用内存
```

```
        Call 签订部门复合框设置                              '重置签订部门复合框
        签订部门.Value = bm                                  '设置签订部门复合框显示值
End Sub
```

11.8.7　新合同与添加按钮代码设计

在合同基本信息管理窗口中可以添加新合同，单击【新合同】按钮与【添加】按钮即可。单击【新合同】按钮后，窗口中各个控件的数据被清除，以便用户输入新合同的基本信息。当用户输入完所有的新合同基本信息后，单击【添加】按钮后，该新合同基本信息将被添加进数据库。下面是【新合同】按钮的单击事件代码：

```
Private Sub 新合同_Click()
        Dim i As Integer
        '清除窗体上各个控件的数据，或将某控件的值设置为默认状态
        For i = 0 To UBound(myArray)
                Me.Controls(myArray(i)).Value = ""                  '清除控件数据
        Next
        合同类别.ListIndex = 0                                       '设置合同类别复合框的值为默认
        签订部门.ListIndex = 0                                       '设置签订部门复合框的值为默认
        签订日期.Value = Format(Date, "yyyy-mm-dd")                  ''设置签订日期文本框的值
        收费合计.Value = ""                                          '清除收费合计文本框数据
        欠费合计.Value = ""                                          '清除欠费合计文本框数据
        ListView1.ListItems.Clear                                    '清除 ListView 控件显示项
        合同记录数目.Caption = "数据库中共有 " & rs.RecordCount & " 条合同记录"    '设置提示标签
        合同号.SetFocus                                              '将光标定位到合同号文本框
End Sub
```

【添加】按钮完成的主要任务是将窗口中输入的新合同基本信息保存到数据库。在完成该任务过程中，程序还需要检测用户是否输入了必要的数据、检测合同号是否有重复。新合同基本信息保存完毕后，还需要重新调用查询、显示合同基本信息过程，更新窗口显示及提示。图 11-33 显示的是该按钮的单击事件过程流程图。

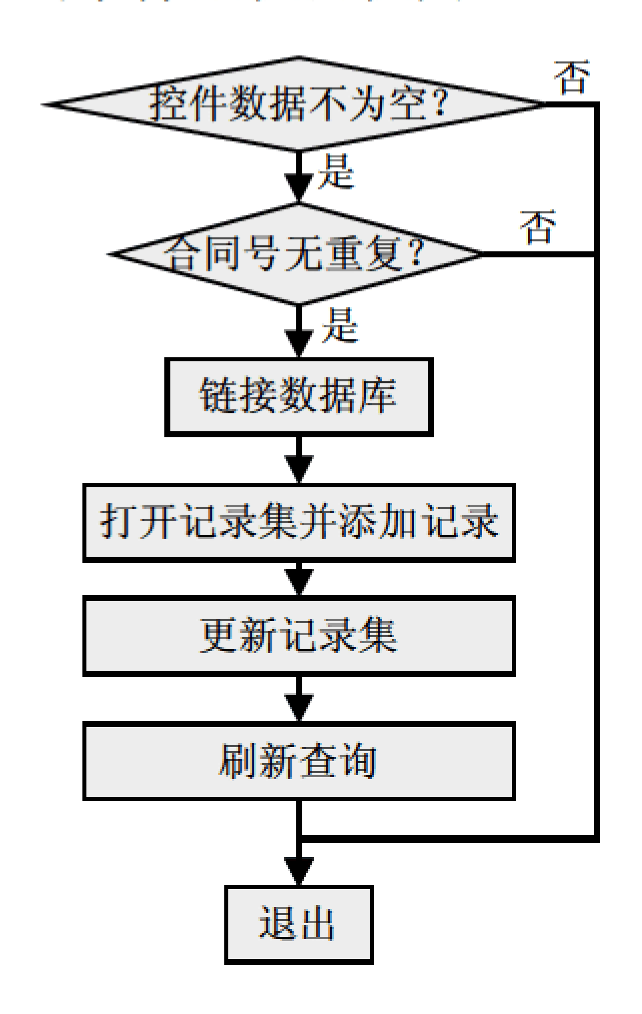


图 11-33　【添加】按钮单击事件过程流程图

以下是【添加】按钮的单击事件过程代码：

```
Private Sub 添加_Click()
    Dim i As Integer
    '判断是否在窗体上输入了必要的合同数据
    For i = 0 To UBound(myArray) -1
        If Me.Controls(myArray(i)).Name <> "备注" Then
            If Me.Controls(myArray(i)).Value = "" Then
                MsgBox Me.Controls(myArray(i)).Name & "不能为空！", vbCritical
                Me.Controls(myArray(i)).SetFocus
                Exit Sub
            End If
        End If
    Next
    If MsgBox("本操作将添加新的合同记录！" & vbCrLf & "是否要添加？", _
        vbQuestion + vbYesNo, "添加记录") = vbNo Then Exit Sub
    '首先判断在数据库中是否存在相同的合同号
    Dim rsNum As New ADODB.Recordset
    SQL = "select 合同号 from 合同基本信息 where 合同号='" & 合同号.Value & "'"
    rsNum.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If rsNum.RecordCount > 0 Then
        MsgBox "在数据库中已经存在有编号为<" & 合同号.Value & ">的合同！" _
            & vbCrLf & "请重新输入合同号！", vbOKOnly + vbCritical, "警告"
        Me.合同号.Value = ""                  '将合同号文本框数据清除
        Me.合同号.SetFocus                    '将焦点移到合同号文字框
        GoTo hhh
    End If
    '准备将窗体上的数据添加到数据库中
    SQL = "select * from 合同基本信息"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    '开始添加数据
    With rs
        .AddNew                              '添加各个字段的数据
        For i = 0 To UBound(myArray)
            If Me.Controls(myArray(i)).Name = "签订日期" _
            Or Me.Controls(myArray(i)).Name = "合同起始日" _
            Or Me.Controls(myArray(i)).Name = "合同终止日" Then
                .Fields(i) = Format(Me.Controls(myArray(i)).Value, "yyyy-mm-dd")
            Else
                .Fields(i) = Me.Controls(myArray(i)).Value
            End If
        Next i
        .Update                              '更新数据表
    End With
    MsgBox "已经成功将新合同数据添加到数据库中！", vbInformation, "添加记录"
    '刷新查询
    Call 查询合同基本信息
    Call 显示合同基本信息
```

```
hhh:
    rsNum.Close
    Set rsNum = Nothing
End Sub
```

11.8.8　修改按钮代码设计

【修改】按钮完成对当前显示合同的基本信息进行修改的任务。当用户需要修改合同基本信息时，在合同基本信息框架中修改好合同的基本信息后，单击该按钮即可。单击【修改】按钮后，程序首先提示是否修改，再单击【确定】按钮后，程序从数据库获取一记录集。该记录集的合同号即为当前显示合同的合同号。然后程序将用户对合同做出的修改保存到数据库中，并更新记录集以确认修改。最后程序调用查询与显示合同基本信息刷新查询。

```
Private Sub 修改_Click()
    If MsgBox("本操作将修改合同号为<" & 合同号.Value & ">的合同记录！" _
        & vbCrLf & "是否要更新？", _
        vbQuestion + vbYesNo, "更新记录") = vbNo Then Exit Sub
    Dim i As Integer
    '准备修改记录
    SQL = "select * from 合同基本信息 where 合同号='" & 合同号.Value & "'"
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    '修改更新记录
    With rs
        For i = 0 To UBound(myArray)
            If Me.Controls(myArray(i)).Name = "签订日期" _
            Or Me.Controls(myArray(i)).Name = "合同起始日" _
            Or Me.Controls(myArray(i)).Name = "合同终止日" Then
                .Fields(i) = Format(Me.Controls(myArray(i)).Value, "yyyy-mm-dd")
            Else
                .Fields(i) = Me.Controls(myArray(i)).Value
            End If
        Next i
        .Update      '更新数据表
    End With
    MsgBox "已经成功将编号为<" & 合同号.Value & ">的合同记录进行了更新！", _
        vbInformation, "更新记录"
    '刷新查询
    Call 查询合同基本信息
    Call 显示合同基本信息
End Sub
```

11.8.9　删除按钮代码设计

【删除】按钮完成删除当前显示合同基本信息记录与合同收费信息的任务。用户需要删除当前合同基本信息与合同收费信息时，只需单击该按钮即可删除当前记录。单击【删除】

按钮后，程序提示是否删除当前显示合同记录。确认后，程序调用 ADO 数据库对象的 Execute 方法执行两条删除记录的 SQL 语句，分别用于删除合同基本信息和合同收费信息。最后程序调用查询、显示合同基本信息和显示合同收费情况刷新查询。以下是该按钮的单击事件代码：

```
Private Sub 删除_Click()
    If MsgBox("本操作将删除编号为<" & 合同号.Value & ">的合同记录！" _
        & vbCrLf & "是否要删除？", _
        vbQuestion + vbYesNo, "删除记录") = vbNo Then Exit Sub
    '删除合同基本信息
    SQL = "delete from 合同基本信息 where 合同号='" & 合同号.Value & "'"
    Set rs = cnn.Execute(SQL)
    '删除合同收费信息
    SQL = "delete from 合同收费信息 where 合同号='" & 合同号.Value & "'"
    Set rs = cnn.Execute(SQL)
    MsgBox "已经成功将编号为<" & 合同号.Value & ">的合同记录删除！", _
        vbInformation, "删除记录"
    '刷新查询
    Call 查询合同基本信息
    Call 显示合同基本信息
    Call 显示合同收费情况
End Sub
```

11.8.10 查询按钮代码设计

【查询】按钮完成查询合同基本信息的任务。该按钮查询的依据是合同号。用户需要查询时，单击该按钮。程序首先清除了窗体上各个控件的数据，然后弹出一输入对话框要求输入查询合同的合同号。输入合同号后，单击【确定】按钮。程序逐条记录在合同基本信息表中查找该合同号，找到后将合同基本信息和收费情况显示在各个控件上。以下是该按钮的单击事件代码：

```
Private Sub 查询_Click()
    Dim myId As String
    Call 新合同_Click                                    '清除窗体上各个控件的数据
    myId = InputBox("请输入合同号：", "合同查询")         '获取查询合同的合同号
    If Len(Trim(myId)) = 0 Then
        MsgBox "没有输入合同号！", vbCritical, "警告"      '提示未输入合同号
        Exit Sub
    End If
    rs.MoveFirst                              '将合同基本信息记录集定位到第一条记录
    For i = 1 To rs.RecordCount
        If rs.Fields("合同号") = myId Then    '找到对应合同号后，显示合同基本信息和收费情况
            Call 显示合同基本信息
            Call 显示合同收费情况
            Exit Sub
        Else
            rs.MoveNext                       '移动到下一条记录行
        End If
```

```
        Next
        MsgBox "没有合同号为<" & myId & ">的合同！", vbCritical, "查询结果"     '显示提示信息
        rs.MoveFirst
End Sub
```

11.8.11　浏览记录按钮组代码设计

在窗体右侧排布的按钮中，有一组按钮是用于浏览记录的。这些按钮包括【第一条】、【下一条】、【上一条】和【最末条】4 个按钮。这些按钮共同完成浏览记录功能，本小节将这些按钮的代码集中起来一起介绍。以下是这些按钮的详细代码：

```
Private Sub 第一条_Click()
        '如果数据表中没有记录，就退出过程
        If rs.BOF And rs.EOF Then Exit Sub
        '如果已经是第一条记录，就退出过程，以免再次单击此按钮时出现错误
        If rs.BOF Then Exit Sub
        '将指针移到第一条记录
        rs.MoveFirst
        '如果已经是第一条记录，就退出过程，以免再次单击此按钮时出现错误
        If rs.BOF Then Exit Sub
        '调用子程序在窗体上显示第一条记录
        Call 显示合同基本信息
        Call 显示合同收费情况
End Sub

Private Sub 下一条_Click()
        '如果数据表中没有记录，就退出过程
        If rs.BOF And rs.EOF Then Exit Sub
        '如果已经是最末一条记录，就退出过程，以免再次单击此按钮时出现错误
        If rs.EOF Then Exit Sub
        '将指针移到下一条记录
        rs.MoveNext
        '如果已经是最末一条记录，就退出过程，以免再次单击此按钮时出现错误
        If rs.EOF Then Exit Sub
        '调用子程序在窗体上显示下一条记录
        Call 显示合同基本信息
        Call 显示合同收费情况
End Sub

Private Sub 上一条_Click()
        '如果数据表中没有记录，就退出过程
        If rs.BOF And rs.EOF Then Exit Sub
        '如果已经是第一条记录，就退出过程，以免再次单击此按钮时出现错误
        If rs.BOF Then Exit Sub
        '将指针移到上一条记录
        rs.MovePrevious
        '如果已经是第一条记录，就退出过程，以免再次单击此按钮时出现错误
        If rs.BOF Then Exit Sub
```

```
    '调用子程序在窗体上显示上一条记录
    Call 显示合同基本信息
    Call 显示合同收费情况
End Sub

Private Sub 最末条_Click()
    '如果数据表中没有记录，就退出过程
    If rs.BOF And rs.EOF Then Exit Sub
    '如果已经是最末一条记录，就退出过程，以免再次单击此按钮时出现错误
    If rs.EOF Then Exit Sub
    '将指针移到最末条记录
    rs.MoveLast
    '如果已经是最末一条记录，就退出过程，以免再次单击此按钮时出现错误
    If rs.EOF Then Exit Sub
    '调用子程序在窗体上显示最末条记录
    Call 显示合同基本信息
    Call 显示合同收费情况
End Sub
```

11.9 合同收费信息管理窗口设计

合同收费信息管理窗口完成对合同收费情况资料的添加、修改和删除等操作。同一个合同允许有多个收费信息。当需要查看某一个合同所有的收费情况信息时，可以在合同收费信息管理窗口的合同收费信息框架中查看，也可以通过该窗口选择对应合同号查看。

11.9.1 窗口界面设计

合同收费信息管理窗口包含的控件数量较合同基本信息管理窗口少，共有 2 个框架控件、5 个标签控件、3 个文本框控件、2 个复合框控件、7 个按钮和 1 个 ListView 控件。该窗口的界面如图 11-34 所示。



图 11-34 合同收费信息管理窗口界面

窗口的布局和合同基本信息管理窗口一致，包括合同收费信息输入区、合同收费情况显示区以及按钮区。表 11-6 列出了本窗体的相关控件（不包含标签控件）。

表 11-6　合同收费信息管理窗口控件列表

控　件　名	控 件 类 型	控 件 说 明
合同收费信息	框架	该框架用于分隔所有显示或设置合同收费信息的控件
合同号	复合框	合同号复合框包含了所有已建立的合同的合同号。通过选择合同号，窗口合同收费情况中的明细项目会随之更新，显示相应合同的收费情况
收费类别	复合框	收费类别复合框包含了所有已定义的合同收费类别。通过单击该控件右侧的【添加类别】按钮，用户可以新建新收费类别
添加类别	按钮	该按钮用于添加新的收费类别。添加完收费类别后，收费类别复合框将会立即刷新
收费日期	文本框	该文本框显示或设置当前合同收费信息的收费日期
收费金额	文本框	该文本框显示或设置当前合同收费信息的收费金额
备注	文本框	该文本框显示或设置当前合同收费信息的备注信息
合同收费情况	框架	该框架用于包含显示合同收费情况的 ListView 控件
ListView1	ListView	该控件用于显示当前合同的所有收费情况
新记录	按钮	该按钮重置合同收费信息框架中的控件数据，以便输入新合同收费信息
添加	按钮	该按钮用于为当前合同添加新收费信息
修改	按钮	该按钮用于修改当前显示合同收费信息
删除	按钮	该按钮用于删除当前显示合同收费信息
查询	按钮	该按钮用于查询满足当前合同号的所有合同收费情况
退出	按钮	该按钮用于退出当前合同收费信息管理窗口

建立该窗口的步骤如下：

（1）在 VBE 环境中依次选择【插入】|【用户窗体】命令建立一个新窗体，并在属性窗口中设置该窗体的名称属性为“合同收费信息管理”，如图 11-35 所示。

（2）在工具箱中选择框架控件。在窗体中插入一个框架。随后在属性窗口中设置该框架的 Caption 属性为“合同收费信息”，名称属性设置为 Frame1，如图 11-36 所示。



图 11-35　合同收费信息管理窗体属性设置

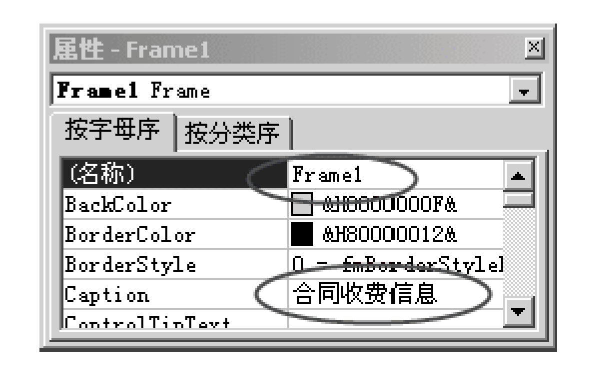


图 11-36　合同收费信息框架属性设置

（3）在工具箱中选择标签控件。在窗体中连续插入 5 个标签控件。随后在属性窗口中设置这些控件的 Caption 属性依次为“合同号”、“收费类别”、“收费日期”、“收费金额”和“备注”，如图 11-37 所示。

（4）在工具箱中选择复合框控件。在“合同号”与“收费类别”标签右侧分别插入一个复合框，随后在属性窗口中设置这两个复合框的名称为“合同号”和“收费类别”。

SelectionMargin 属性都设置为 False，如图 11-38 所示。





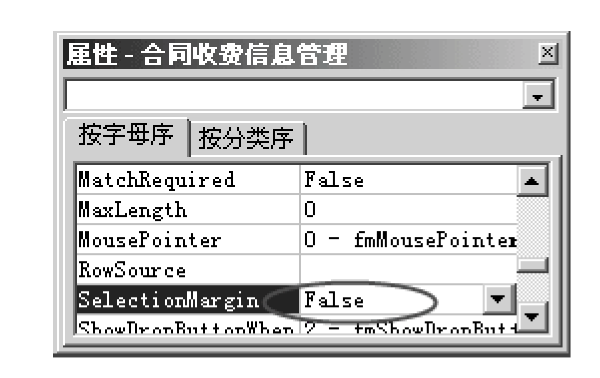
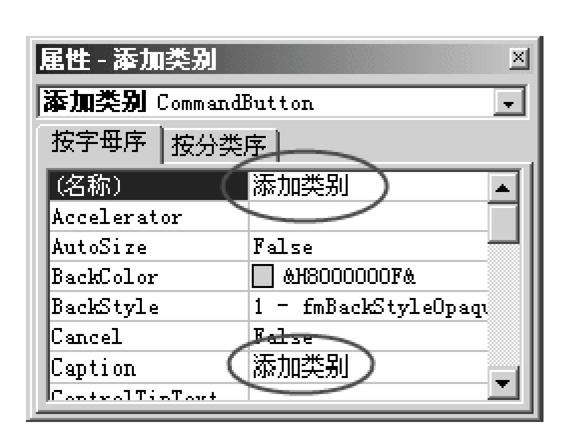


图 11-37　合同收费信息管理窗体设计效果　　图 11-38　设置复合框控件的 SelectionMargin 属性

（5）在工具箱中选择按钮控件。在“收费类别”复合框右侧插入一个按钮。随后在属性窗口中设置该按钮的 Caption 属性为“添加类别”，名称属性为“添加类别”，如图 11-39 所示。

（6）在工具箱中选择文本框控件。在“收费日期”、“收费金额”和“备注”标签右侧分别插入文本框。随后在属性窗口中设置这些文本框的名称属性依次为“收费日期”、“收费金额”、“备注”，SelectionMargin 属性都设置为 False。

（7）在工具箱中选择框架控件。在窗体上插入一个框架。随后在属性窗口中设置该框架的 Caption 属性为“合同收费情况”，名称属性设置为 Frame2，如图 11-40 所示。



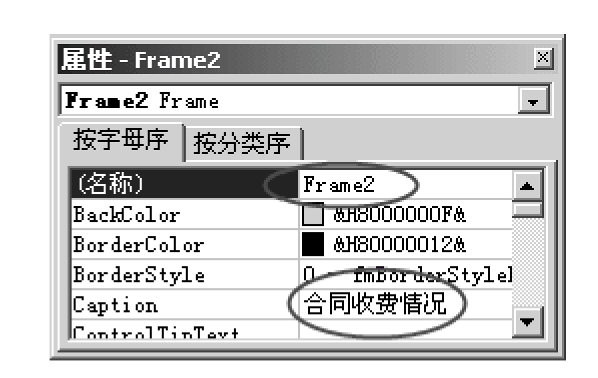


图 11-39　添加类别按钮属性设置　　图 11-40　合同收费情况框架属性设置

（8）在工具箱中选择 ListView 控件。在“合同收费情况”框架中插入一个 ListView 控件，随后在属性窗口中设置该控件的名称属性为 ListView1。

（9）在工具箱中选择按钮控件。在“合同收费信息”框架右侧连续插入 6 个按钮，随后在属性窗口中设置这些按钮的 Caption 属性依次为“新记录”、“添加”、“修改”、“删除”、“查询”和“退出”。设置这些控件的名称属性与其 Caption 属性一致。

11.9.2　窗口初始化与关闭事件代码设计

本小节讲述的是该窗体的初始化与关闭事件代码。窗口初始化过程完成的工作主要是设置窗体控件状态、链接数据库、设置复合框、显示合同收费信息和显示合同收费情况。图 11-41

显示的是该窗体初始化过程的流程图。

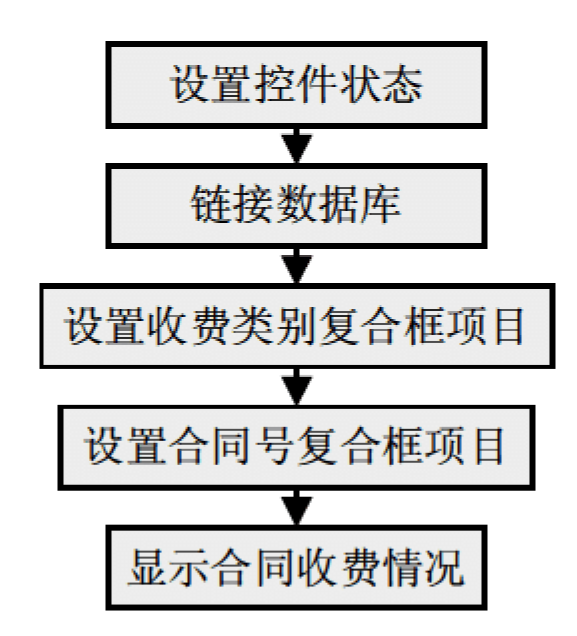


图 11-41　合同收费信息管理窗口初始化代码流程图

该窗口的初始化代码如下：

```
Private Sub UserForm_Initialize()
    Dim mydata As String
    Dim SQL As String
    Dim i As Integer
    '指定数据库
    mydata = ThisWorkbook.Path & "\合同管理.mdb"
    '设置窗体控件组（也是数据表的各个字段组）
    myArray = Array("合同号", "收费类别", "收费日期", "收费金额", "备注")
    '设置窗体控件的背景颜色
    For i = 0 To UBound(myArray)
        Me.Controls(myArray(i)).BackColor = &HE0E0E0        '指定控件背景色
    Next
    '建立与数据库的连接
    Set cnn = New ADODB.Connection
    With cnn
        .Provider = "microsoft.jet.oledb.4.0"               '指定链接的 Provider 属性
        .Open mydata                                        '开启链接
    End With
    '调用子程序，为收费类别复合框设置项目
    Call 收费类别复合框设置
    '从“合同基本信息”中查询合同号，设置给“合同号”复合框
    Dim rsx As New ADODB.Recordset
    SQL = "select 合同号 from 合同基本信息"
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic      '打开记录集
    With 合同号
        .Clear                                              '清除合同号复合框记录
        For i = 1 To rsx.RecordCount
            .AddItem rsx.Fields("合同号")                   '添加合同号记录
            rsx.MoveNext                                    '将记录向下移动一条
        Next
    End With
    合同号.ListIndex = 0                                    '指定合同号复合框默认值
    rsx.Close                                               '关闭记录集
    Set rsx = Nothing
    '调用子程序，查询并显示合同收费明细信息
    Call 查询合同收费明细                                   '调用查询合同收费明细过程
```

```
    Call 显示合同收费明细                                   '调用显示合同收费明细过程
End Sub

Private Sub 退出_Click()
    cnn.Close                                             '关闭数据库链接
    Set rs = Nothing                                      '清除记录集内存空间
    Set cnn = Nothing                                     '清除链接内存空间
    Unload 合同收费信息管理                                 '卸载窗口
End Sub
```

11.9.3 复合框设置代码设计

在合同收费信息框架中，包含了合同号复合框和收费类别复合框两个复合框，其中设置合同号复合框的过程已经在窗口初始化代码中完成。因而在这里讲述的只是合同号复合框改变事件和收费类别复合框设置项目。

当合同号复合框的输入内容改变时，程序首先将各个控件的显示数据清除，然后查询当前输入合同的收费明细并显示出来，最后把合同号、收费类别、收费日期以及收费金额保存在公共变量中。

设置收费类别复合框的过程是：首先从收费类别信息表中获取记录集，然后将该记录集中的收费类别字段中保存的数据写入到收费类别复合框的项目中。

```
Private Sub 合同号_Change()
    Dim i As Integer
    For i = 1 To UBound(myArray)
        Me.Controls(myArray(i)).Value = ""                '清除控件数据
    Next
    Call 查询合同收费明细                                   '调用查询合同收费明细过程
    Call 显示合同收费信息                                   '调用显示合同收费信息过程
    Call 显示合同收费明细                                   '调用显示合同收费明细过程
    hth = 合同号.Value                                     '保存合同号
    sflb = 收费类别.Value                                  '保存收费类别
    sfrq = Format(收费日期.Value, "yyyy-mm-dd")            '保存收费日期
    sfje = Val(收费金额.Value)                             '保存收费金额
End Sub

Public Sub 收费类别复合框设置()
    Dim rsx As New ADODB.Recordset
    rsx.Open "收费类别信息", cnn, adOpenKeyset, adLockOptimistic     '获取收费类别记录集
    With 收费类别
        .Clear                                            '清空收费类被复合框项目
        For i = 1 To rsx.RecordCount
            .AddItem rsx.Fields(0)                        '添加收费类被项目
            rsx.MoveNext                                  '将记录集指针移到下一条
        Next
    End With
    rsx.Close                                             '关闭记录集
    Set rsx = Nothing                                     '清除记录集占用内存空间
End Sub
```

11.9.4 查询、显示合同收费信息代码设计

前面的窗口初始化代码与合同号复合框改变事件中都涉及到了查询、显示合同收费信息的自定义过程。这些自定义过程完成获取合同收费信息记录集与显示合同收费信息的工作。各个过程的功能描述如下：

- ❑ 查询合同收费明细过程：根据用户输入的合同号，在数据库中查询对应合同号的所有收费情况。其查询结果被保存在一公共记录集对象中，该记录集对象将在显示合同收费明细过程中被调用。
- ❑ 显示合同收费信息过程：该过程用于将查询合同收费信息记录集的第一条记录显示在合同收费信息框架的几个控件中。
- ❑ 显示合同收费明细过程：该过程显示查询合同收费信息记录集的所有记录到 ListView 控件中。

以下是这 3 个过程的代码解释：

```
Public Sub 查询合同收费明细()
    Dim SQL As String
    SQL = "select * from 合同收费信息 where 合同号='" & 合同号.Value & "'"        '设置查询字符串
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic                          '打开记录集
End Sub

Public Sub 显示合同收费信息()
    On Error Resume Next
    Dim i As Integer
    '显示合同收费的第一条信息
    For i = 0 To UBound(myArray)
        If IsNull(rs.Fields(i)) Then
            Me.Controls(myArray(i)).Value = ""              '当记录集为空时，清空控件数据
        Else
            Me.Controls(myArray(i)).Value = rs.Fields(i)    '记录集不为空时，显示对应字段到控件中
        End If
    Next i
End Sub

Public Sub 显示合同收费明细()
    On Error Resume Next
    With ListView1
        '设置 ListView1 的标题、显示类型、整行选择和网格线属性
        .ColumnHeaders.Clear                        '清除 ListView 控件标题
        .ListItems.Clear                            '清除 ListView 控件数据项
        .View = lvwReport                           '设置 ListView 控件显示模式
        .FullRowSelect = True                       '允许整行选择
        .Gridlines = True                           '显示网格线
        '为 ListView1 设置标题
        For i = 0 To rs.Fields.Count -1
```

```
            .ColumnHeaders.Add , , rs.Fields(i).Name          '添加控件标题
        Next i
        '为 ListView1 设置各行数据
        For i = 1 To rs.RecordCount
            .ListItems.Add , , rs.Fields(0).Value             '为 ListView 控件添加项目
            For j = 1 To rs.Fields.Count -1
                .ListItems(i).SubItems(j) = rs.Fields(j).Value  '指定新项目的子项数据
            Next j
            rs.MoveNext                                       '将记录集指向下一条
        Next
    End With
    rs.MoveFirst                                              '将记录集指针指向第一条记录
End Sub
```

11.9.5　添加类别按钮代码设计

【添加类别】按钮用于新增类别。在类别复合框中输入类别后，单击该按钮，新类别将会被保存到收费类别信息表中。在执行该过程中，程序首先检测用户是否已输入收费类别，确认以后程序打开到费类别信息表的记录集。当记录集中没有该收费类别时，程序将该新收费类别添加进收费类别信息表中。如图 11-42 所示的是该过程的流程图。

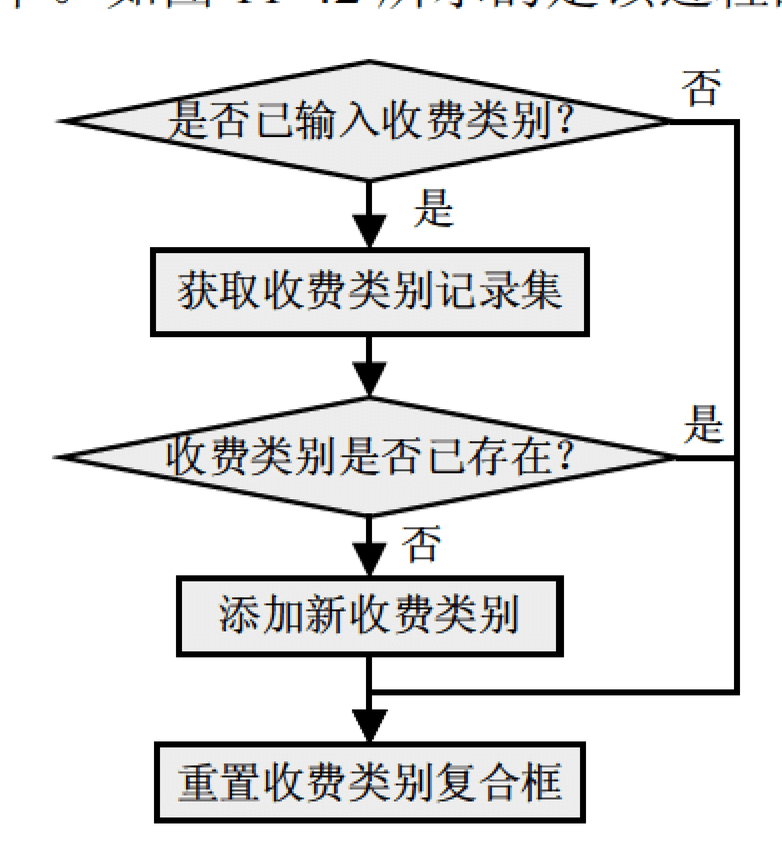


图 11-42　添加合同收费类别流程图

以下是该按钮的单击事件代码解释：

```
Private Sub 添加类别_Click()
    Dim lb As String
    lb = 收费类别.Value
    If Len(Trim(lb)) = 0 Then
        MsgBox "没有输入收费类别名称！不能添加！", vbCritical, "警告"      '提示未输入收费类别
        Exit Sub
    End If
    Dim rsx As New ADODB.Recordset
    Dim SQL As String
    SQL = "select * from 收费类别信息 where 收费类别='" & lb & "'"          '生成查询字符串
    rsx.Open SQL, cnn, adOpenKeyset, adLockOptimistic     '获取指定收费类被的收费类别记录集
```

```
    If rsx.BOF And rsx.EOF Then
        '当记录集为空时，将新输入的收费类别保存到收费类别表中
        rsx.AddNew                                              '添加新记录
        rsx.Fields(0) = lb                                      '保存新收费类别
        rsx.Update                                              '更新记录集
        MsgBox "添加完毕！", vbInformation, "添加收费类别"            '提示保存成功
    Else
        MsgBox "已经存在了同名的收费类别名称！", vbCritical, "警告"    '提示收费类别已经存在
    End If
    rsx.Close                                                   '关闭记录集
    Set rsx = Nothing                                           '清空记录集占用内存
    Call 收费类别复合框设置                                       '重置收费类别复合框
    收费类别.Value = lb                                          '设置收费类别复合框的显示值
End Sub
```

11.9.6 新记录与添加按钮代码设计

当用户需要通过合同收费信息管理窗口添加新的合同收费信息时，需要完成两步操作。首先用户需要单击【新记录】按钮来清除合同收费信息与合同收费情况框架中部分控件的内容，然后用户可以在这些被清空的输入控件中输入新的合同收费信息，最后单击【添加】按钮让程序自动完成新合同信息添加工作。该小节将两个按钮的代码放置在一起，正是因为这两个按钮完成的工作是同一个任务的两个步骤。

其中，【添加】按钮的单击事件的代码比较复杂。单击【添加】按钮时，程序首先检查用户是否在窗口输入了必要的合同收费信息。满足条件后程序从数据库中获取合同总金额与合同收费合计信息。然后程序比较这两个数据，判断是否需要登记新的合同收费信息。最后程序将刷新窗口的显示数据，以显示用户添加的合同收费数据。如图 11-43 所示是【添加】按钮单击事件的流程图。

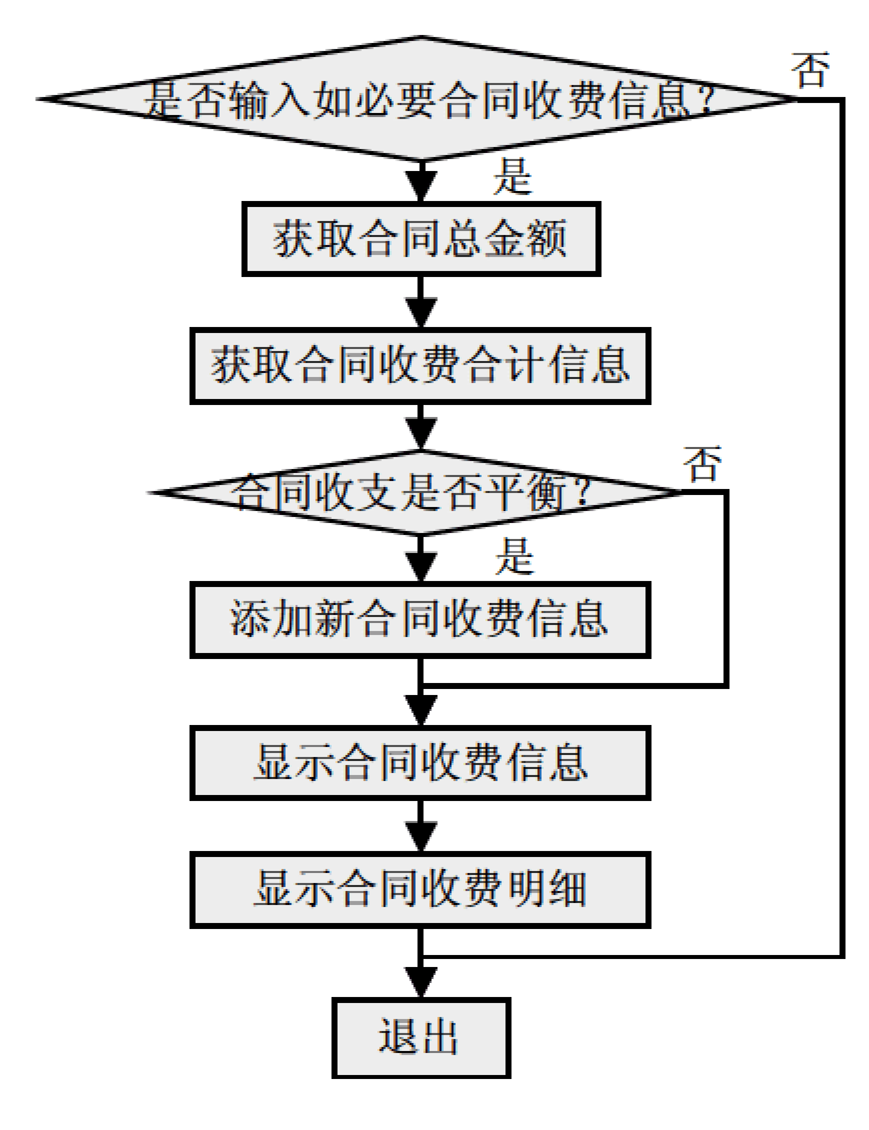


图 11-43 【添加】按钮单击事件流程图

以下是这两个按钮的单击事件代码解释：

```
Private Sub 新记录_Click()
    On Error Resume Next
    Dim i As Integer
    For i = 0 To UBound(myArray)
        Me.Controls(myArray(i)).Value = ""                     '清空控件显示数据
    Next
    收费类别.ListIndex = 0                                     '设置收费类别复合框默认显示值
    收费日期.Value = Format(Date, "yyyy-mm-dd")                '设置收费日期
    合同号.SetFocus                                            '设置鼠标焦点
End Sub

Private Sub 添加_Click()
    Dim i As Integer
    Dim TotalMoney As Currency, TotalIncome As Currency
    Dim rsCurrency As ADODB.Recordset
    '判断是否在窗体上输入了必要的合同收费数据
    For i = 0 To UBound(myArray)
        If Me.Controls(myArray(i)).Name <> "备注" Then          '备注项不需检查
            If Me.Controls(myArray(i)).Value = "" Then
                MsgBox Me.Controls(myArray(i)).Name & "不能为空！", vbCritical
                                                               '提示有未输入数据时
                Me.Controls(myArray(i)).SetFocus               '定位未输入数据控件
                Exit Sub                                       '退出过程
            End If
        End If
    Next
    If MsgBox("本操作将添加新的合同收费记录！" & vbCrLf & "是否要添加？", _
        vbQuestion + vbYesNo, "添加记录") = vbNo Then Exit Sub  '提示是否添加新合同收费信息
    '首先判断在数据库中该合同的收费与欠费是否已经平衡
    '查询该合同的总金额
    Set rsCurrency = New ADODB.Recordset
    SQL = "select 合同金额 from 合同基本信息 where 合同号='" & 合同号.Value & "'"
                                                               '生成查询字符串
    rsCurrency.Open SQL, cnn, adOpenKeyset, adLockOptimistic   '获取合同金额记录集
    TotalMoney = rsCurrency.Fields("合同金额")                 '保存合同金额
    '查询该合同的收费合计
    Set rsCurrency = New ADODB.Recordset
    SQL = "select sum(收费金额) as aa from 合同收费信息 " _
        & "where 合同号='" & 合同号.Value & "'"                '生成查询字符串
    rsCurrency.Open SQL, cnn, adOpenKeyset, adLockOptimistic   '获取合同收费金额记录集
    If IsNull(rsCurrency.Fields!aa) Then
        TotalIncome = 0                          '当没有收费金额时，保存合同收费金额为 0
    Else
        TotalIncome = rsCurrency.Fields!aa       '有收费金额时，直接保存该合同收费金额
    End If
    '判断合同费用是否已经结清
    If TotalIncome >= TotalMoney Then            '当合同收费金额未超过合同总金额时，可以添加
                                                  新合同收费金额
        MsgBox "该合同的各项费用已经结清！不能再添加记录！", vbCritical, "警告"
```

```
    Else
        '准备将窗体上的数据添加到数据库中
        SQL = "select * from  合同收费信息"                    '生成查询字符串
        Set rs = New ADODB.Recordset
        rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic      '获取合同收费信息记录集
        '开始添加数据
        With rs
            .AddNew                                           '添加新数据
            For i = 0 To UBound(myArray)
                If Me.Controls(myArray(i)).Name = "收费日期" Then
                    .Fields(i) = Format(Me.Controls(myArray(i)).Value, "yyyy-mm-dd")
                                                              '设置收费日期字段数据
                Else
                    .Fields(i) = Me.Controls(myArray(i)).Value     '设置其他字段数据
                End If
            Next
            .Update                                           '更新数据表
        End With
        MsgBox "已经成功将该合同的收费数据添加到数据库中！", vbInformation, "添加记录"
        '刷新显示
        Call  显示合同收费信息                                 '调用显示合同收费信息过程
        Call  查询合同收费明细                                 '调用查询合同收费明细过程
        Call  显示合同收费明细                                 '调用显示合同收费明细过程
    End If
    rsCurrency.Close                                          '关闭记录集
    Set rsCurrency = Nothing                                  '清空记录集占用内存
End Sub
```

11.9.7　修改按钮代码设计

用户在窗口中单击【修改】按钮后，程序首先提示用户是否进行修改操作，当被用户确认后，程序生成更新数据库查询字符串，然后通过记录集对象执行该更新记录集查询字符串，最后程序将该窗口刷新以体现用户做出的修改。如图 11-44 所示的是该按钮的单击事件流程图。

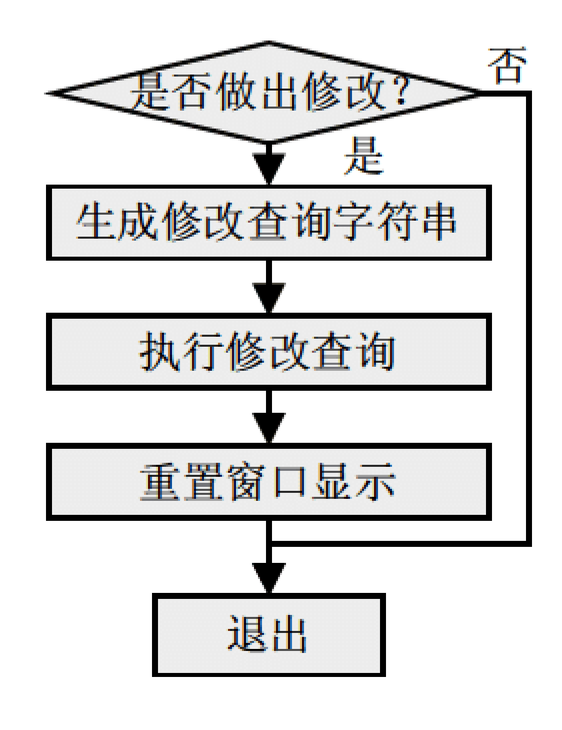


图 11-44　修改合同收费信息流程图

以下是该按钮单击事件过程代码解释：

```
Private Sub 修改_Click()
    If MsgBox("本操作将修改合同号为<" & hth & ">的合同收费记录！" _
        & vbCrLf & "是否要更新？", _
        vbQuestion + vbYesNo, "更新记录") = vbNo Then Exit Sub      '提示是否修改合同收费信息
    Dim i As Integer
    '修改更新记录
    SQL = "update 合同收费信息 set " _
        & "合同号='" & hth & "'," _
        & "收费类别='" & 收费类别.Value & "'," _
        & "收费日期=#" & Format(收费日期.Value, "yyyy-mm-dd") & "#," _
        & "收费金额=" & Val(收费金额) & "," _
        & "备注='" & Trim(备注.Value) & "' " _
        & " where 合同号='" & hth & "'" _
        & " and 收费类别='" & sflb & "'" _
        & " and 收费日期=#" & sfrq & "#" _
        & " and 收费金额=" & sfje                                   '生成更新查询字符串
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic                 '执行更新查询
    MsgBox "已经成功将编号为<" & hth & ">的合同收费记录进行了更新！", _
        vbInformation, "更新记录"                                    '显示更新成功提示
    '刷新显示
    Call 查询合同收费明细                                            '调用查询合同收费明细过程
    Call 显示合同收费明细                                            '调用显示合同收费明细过程
End Sub
```

11.9.8　删除按钮代码设计

【删除】按钮执行的操作与【更新】按钮类似，唯一不同之处在于该查询字符串是一个删除查询字符串而已。该按钮单击事件的流程图如图 11-45 所示。

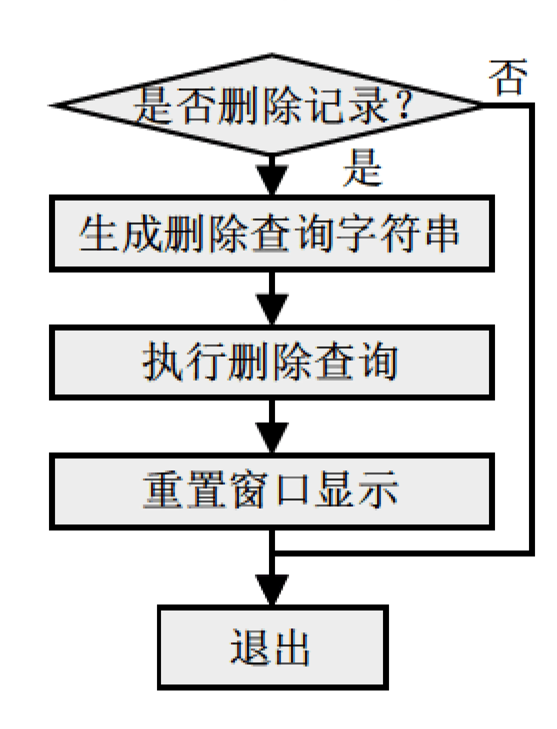


图 11-45　【删除】按钮单击事件流程图

以下是该单击事件的代码解释：

```
Private Sub 删除_Click()
    If MsgBox("本操作将删除编号为<" & hth & ">的合同收费记录！" _
```

```
            & vbCrLf & "是否要删除？", _                              '提示是否删除合同收费记录
            vbQuestion + vbYesNo, "删除记录") = vbNo Then Exit Sub
    SQL = "delete from 合同收费信息 where 合同号='" & hth & "'" _
            & " and 收费类别='" & sflb & "'" _
            & " and 收费日期=#" & sfrq & "#" _
            & " and 收费金额=" & sfje                               '生成删除查询字符串
    Set rs = cnn.Execute(SQL)                                       '运行删除查询
    MsgBox "已经成功将编号为<" & hth & ">的合同收费记录删除！", _
            vbInformation, "删除记录"                               '提示删除成功信息
    '刷新显示
    Call 查询合同收费明细                                           '调用查询合同收费明细过程
    Call 显示合同收费明细                                           '调用显示合同收费明细过程
End Sub
```

11.9.9　查询按钮代码设计

本窗口中的查询功能只能完成对对应合同号的查询工作。当用户单击【查询】按钮时，程序首先将清空所有控件的显示数据，然后弹出一个输入对话框要求用户输入查询合同号。用户完成输入后，程序在合同收费信息表中查询满足指定合同号的记录集。当该记录集有记录时，程序将该记录信息显示到窗口中作为用户的查询结果。图 11-46 是该按钮单击事件的流程图。

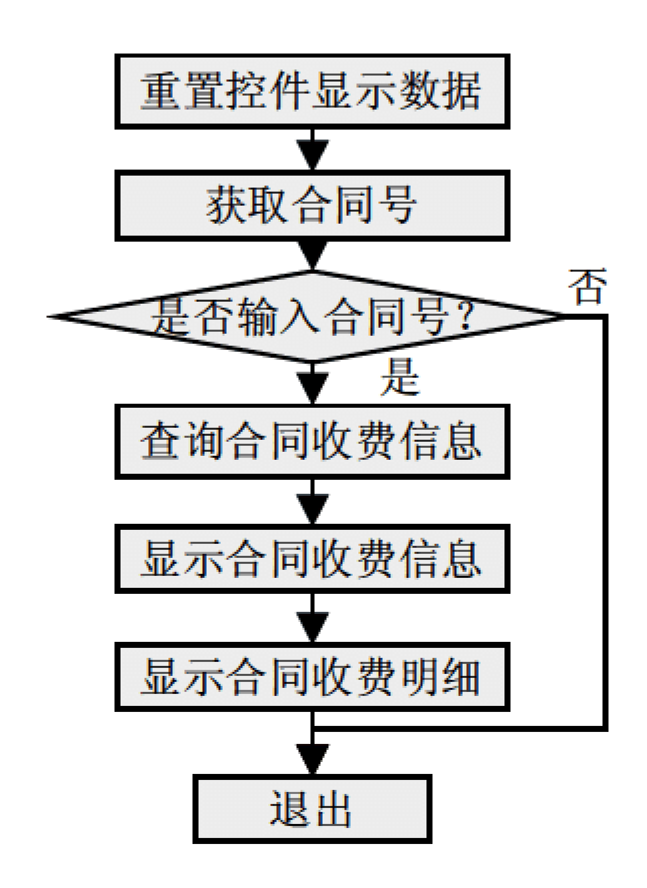


图 11-46　【查询】按钮单击事件流程图

以下是该按钮单击事件的代码解释：

```
Private Sub 查询_Click()
    Dim myId As String
    Dim SQL As String
    Dim i As Integer
    Dim rsSerch As New ADODB.Recordset
    For i = 0 To UBound(myArray)
        Me.Controls(myArray(i)).Value = ""                          '置空控件数据
    Next
    ListView1.ListItems.Clear                                       '清空 ListView 控件显示项目
```

```
    myId = InputBox("请输入合同号：", "合同查询")                     '获取查询合同号
    If Len(Trim(myId)) = 0 Then                                       '检查用户是否输入合同号
        MsgBox "没有输入合同号！", vbCritical, "警告"                  '提示用户未输入合同号
        Exit Sub
    End If
    SQL = "select * from 合同收费信息 where 合同号='" & myId & "'"     '生成查询字符串
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic                  '获取自定合同号的合同收费
                                                                       信息记录集
    If rs.BOF And rs.EOF Then                                         '检测记录集是否有记录
        MsgBox "没有合同号为<" & myId & ">的合同收费信息！", vbCritical, "查询结果"
                                                                      '提示无此合同号
    Else
        Call 显示合同收费信息                                         '调用显示合同收费信息过程
        Call 显示合同收费明细                                         '调用显示合同收费明细过程
    End If
End Sub
```

11.9.10 ListView 控件项目单击事件代码设计

在合同收费情况框架中的 ListView 控件中选择项目时，需要将该项目的数据显示在合同收费信息框架中。程序通过调用查询合同收费明细过程得到该合同号的合同收费信息记录集。为了定位到用户选择项目对应的记录，程序首先将记录指针定位到首条记录，然后通过记录集的 AbsolutePosition 属性定位到选定项目的索引号记录，最后程序将该记录的数据显示到合同收费信息框架的对应控件中，其中有部分数据需要保存到公共变量中以供其他功能调用。

```
Private Sub ListView1_ItemClick(ByVal Item As MSComctlLib.ListItem)
    Dim i As Integer
    Call 查询合同收费明细     '调用查询合同收费明细过程，获取对应合同的合同收费信息记录集
    rs.MoveFirst              '将记录指针定位到首条记录
    rs.AbsolutePosition = Item.Index              '将记录指针定位到用户选定项目对应记录行
    For i = 0 To UBound(myArray)
        If IsNull(rs.Fields(i)) Then
            Me.Controls(myArray(i)).Value = ""    '当记录字段为空时，将对应控件数据显示为空
        Else
            Me.Controls(myArray(i)).Value = rs.Fields(i)      '显示字段的数据到对应控件中
        End If
    Next
    hth = 合同号.Value                                         '保存合同号
    sflb = 收费类别.Value                                      '保存收费类别
    sfrq = Format(收费日期.Value, "yyyy-mm-dd")                '保存收费日期
    sfje = Val(收费金额.Value)                                 '保存收费金额
End Sub
```

11.10　合同信息查询与导出窗口设计

在合同信息查询与导出窗口中，用户可以完成对已建立信息合同的查询与导出工作。该窗口的查询条件设置比合同基本信息管理窗口和合同收费信息管理窗口全面。

11.10.1　窗口界面设计

合同信息查询与导出窗口中包含的数量众多，包含了 3 个框架控件、6 个标签控件、5 个复合框控件、4 个按钮控件和 1 个 ListView 控件。该窗口的界面如图 11-47 所示。



图 11-47　合同信息查询与导出窗口界面

值得注意的是，在设置查询条件框架包含的 4 个复合框中，其中一个在程序运行当中可能会被隐藏。当用户设置查询运算符为 between 时，最后一个复合框才会被显示出来。合同信息查询与导出窗口控件列表见表 11-7。

表 11-7　合同信息查询与导出窗口控件列表

控件名称	控件类型	控件说明
Frame1	框架	该框架将选择查询信息种类和设置查询条件功能区域与其他部分划分开来。它内部还包含了另外一个框架，用于设置查询条件。框架 Caption 属性为“选择要查询信息种类和设置查询条件”
信息种类	复合框	该复合框选择项目用于区分筛选条件的查询信息种类。有合同基本信息与合同收费信息两种
Frame3	框架	该框架内部包含的控件用于设置查询条件。框架 Caption 属性为“设置查询条件”
查询项目	复合框	该复合框选择项目用于设置所需查询的项目名称，包括合同号、项目名称、委托单位等
运算符	复合框	该复合框选择项目用于设置查询条件的运算方式，包括=、>、<等
条件值 1	复合框	该复合框用于设置查询项目满足的条件，该复合框始终可见
条件值 2	复合框	该复合框用于设置查询项目满足的条件。只有运算符为 between 时，该复合框才可见

续表

控件名称	控件类型	控件说明
Frame2	框架	该框架包含显示查询结果的 ListView 控件
Listveiw1	ListView	该控件用于显示查询结果
重设条件	按钮	清空窗口输入控件，以便输入新的查询条件
开始查询	按钮	单击该按钮将使用当前查询条件查询数据库，并将查询结果显示在 ListView 控件中
数据导出	按钮	单击该按钮将把已查询并显示在 ListView 控件中的记录导出到一新工作簿中
关闭窗体	按钮	用于关闭窗口

建立该窗口的步骤如下：

（1）在 VBE 开发环境中依次选择【插入】|【用户窗体】命令。在属性窗口中设置新插入窗体的名称属性为“合同信息查询与导出”，如图 11-48 所示。



图 11-48　合同信息查询与导出窗体属性设置

（2）在工具箱中选择框架控件。在窗口中连续插入 3 个框架，将第二个框架置于第一个框架的中下部。随后在属性窗口中设置 3 个框架的 Caption 属性依次为“选择要查询的信息种类和设置查询条件”、“设置查询条件”和“查询结果显示”，如图 11-49 所示。

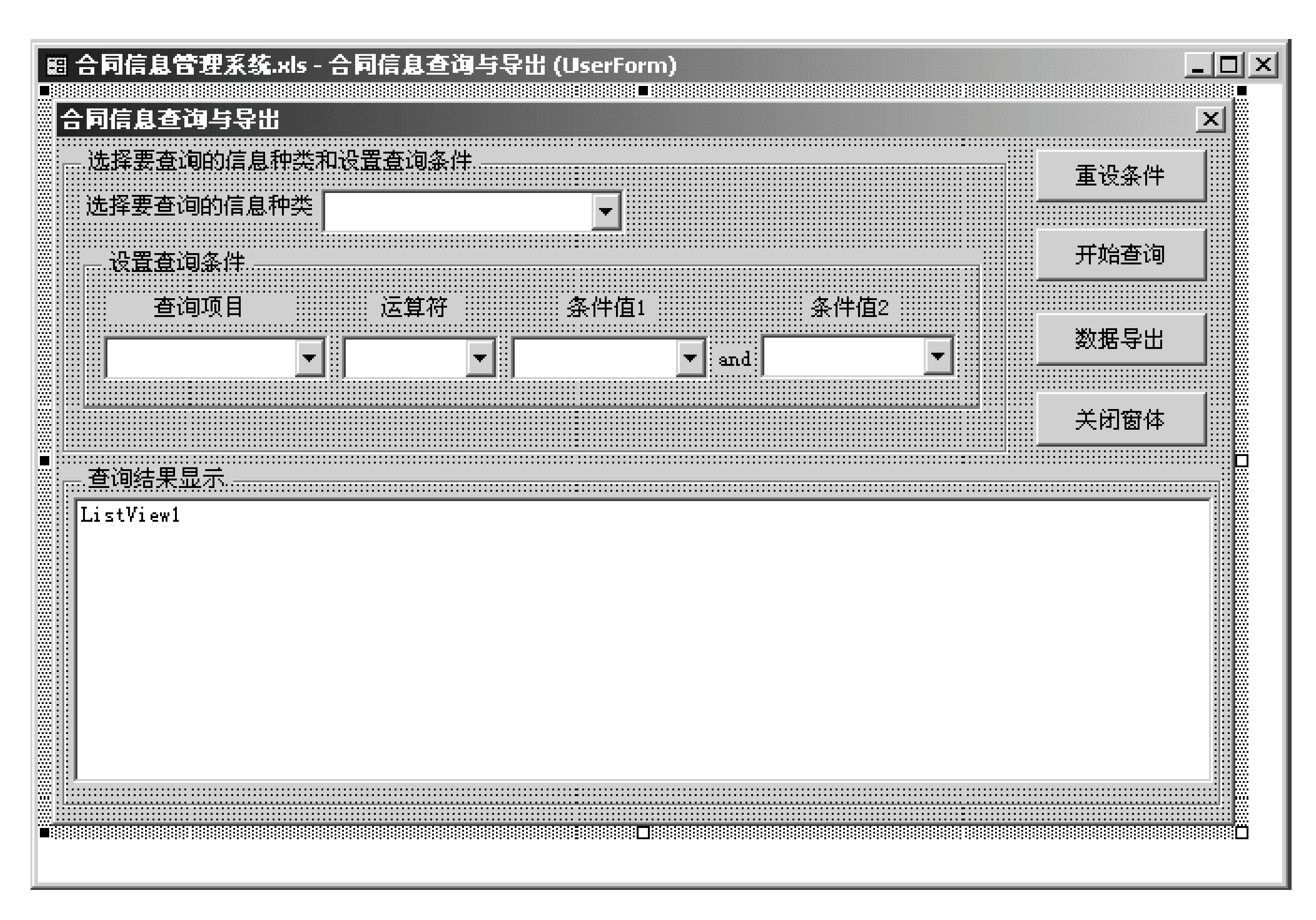


图 11-49　合同信息查询与导出窗体设计效果

（3）在工具箱中选择标签控件。在第一个框架中的上部插入一个标签，然后在第二个框架中的上部依次插入 5 个标签。随后在属性窗口中设置这 6 个标签的 Caption 属性依次为“选择要查询的信息种类”、“查询项目”、“运算符”、“条件值 1”、“条件值 2”和“and”。

（4）在工具箱中选择复合框控件。在第一个框架中刚插入标签的右边插入一个复合框。然后在第二个框架中的下部依次插入 4 个复合框。随后在属性窗口中设置这 5 个复合框的名称属性依次为："信息种类"、"查询项目"、"运算符"、"条件值 1"和"条件值 2"。将 SelectionMargin 属性都设置为 False。

（5）在工具箱中选择 ListView 控件。在窗口的第三个框架中插入一个 ListView 控件。随后在属性窗口中设置该控件的名称属性为 ListView1，如图 11-50 所示。



图 11-50　ListView 控件属性设置

（6）在工具箱中选择按钮控件。在窗口的右侧连续插入 4 个按钮控件。随后在属性窗口中设置这些按钮的名称属性依次为"重设条件"、"开始查询"、"数据导出"和"关闭窗体"。然后设置这些按钮的 Caption 属性和名称一致即可。

11.10.2　窗口初始化与关闭事件代码

合同信息查询与导出窗口初始化时，需要完成 3 个任务，分别是链接数据库、设置信息种类复合框项目、设置运算符复合框项目。该过程的流程十分简单，只是由于过程中需要添加的项目较多造成过程代码较长，这里不再列出该过程的流程图。在窗口关闭时，程序只需要清理一些变量然后关闭窗口。

以下是这两个过程的代码解释：

```
Private Sub UserForm_Initialize()
    Dim mydata As String
    '指定数据库
    mydata = ThisWorkbook.Path & "\合同管理.mdb"          '指定数据库所在位置
    '建立与数据库的连接
    Set cnn = New ADODB.Connection
    With cnn
        .Provider = "microsoft.jet.oledb.4.0"              '指定数据库链接的 Provider 属性
        .Open mydata                                        '开启数据库链接
    End With
    '为信息种类复合框设置项目
    With 信息种类
        .AddItem "合同基本信息"                              '添加信息种类第一个项目
        .AddItem "合同收费信息"                              '添加信息种类第二个项目
    End With
    信息种类.ListIndex = 0                                  '指定信息种类复合框默认显示值
    '为运算符复合框设置项目
    With 运算符
        .AddItem "="                                        '添加运算符第一个项目
        .AddItem ">"                                        '添加运算符第二个项目
        .AddItem "<"                                        '添加运算符第三个项目
        .AddItem ">="                                       '添加运算符第四个项目
        .AddItem "<="                                       '添加运算符第五个项目
        .AddItem "<>"                                       '添加运算符第六个项目
```

```
            .AddItem "like"                          '添加运算符第七个项目
            .AddItem "between"                       '添加运算符第八个项目
        End With
        运算符.ListIndex = 0                         '指定运算符复合框默认显示值
End Sub

Private Sub 关闭窗体_Click()
        cnn.Close                                    '关闭数据库链接
        Set rs = Nothing                             '清理记录集对象
        Set cnn = Nothing                            '清理数据库连接对象
        Unload 合同信息查询与导出                    '卸载窗口
End Sub
```

11.10.3 复合框设置代码设计

在窗口中包含了很多的复合框，但是只有其中的 3 个复合框需要编写改变事件代码。这 3 个复合框分别是信息种类复合框、查询项目复合框和运算符复合框。这些复合框的改变事件的功能如下：

- ❑ 信息种类复合框改变事件：根据用户选择的信息种类，从数据库中获取对应表的所有字段信息。这些字段将作为查询项复合框的项目。
- ❑ 查询项目复合框改变事件：根据用户选择查询项目，从数据库中获取当前查询条件字段数据，然后将这些数据添加到条件 1 和条件 2 复合框中，同时清除在 ListView 控件中的数据显示。
- ❑ 运算符复合框改变事件：当用户选择了 between 条件时，两个设置条件的复合框都应该被显示出来。当用户选择其他的查询条件时，应该将第二个条件设置复合框隐藏。

以下是这 3 个复合框的改变事件代码解释：

```
Private Sub 信息种类_Change()
        On Error Resume Next
        With 查询项目
            .Clear                                          '清除查询项目复合框所有项目
            Set rs = New ADODB.Recordset
            SQL = "select * from " & 信息种类.Value         '生成查询字符串
            rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic '获取查询记录集
            For i = 0 To rs.Fields.Count -1
                .AddItem rs.Fields(i).Name    '将记录集中所有的字段名称作为查询项目复合框的项目
            Next
        End With
        查询项目.ListIndex = 0                '设置查询项目的默认显示值
        Call 清除显示信息                     '清除 ListView 控件中的显示
End Sub

Private Sub 查询项目_Change()
        On Error Resume Next
        Set rs = New ADODB.Recordset
        SQL = "select distinct " & 查询项目.Value & " from " & 信息种类.Value     '生成查询字符串
```

```
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic        '获取查询记录集
    条件值 1.Clear                                           '清除条件值 1 复合框所有项目数据
    条件值 2.Clear                                           '清除条件值 2 复合框所有项目数据
    For i = 1 To rs.RecordCount
        条件值 1.AddItem rs.Fields(查询项目.Value)            '为条件值 1 复合框添加项目
        条件值 2.AddItem rs.Fields(查询项目.Value)            '为条件值 2 复合框添加项目
        rs.MoveNext                                          '将记录集指针移到下一条指针
    Next
    条件值 1.ListIndex = 0                                   '设置条件值 1 复合框默认值
    条件值 2.ListIndex = 0                                   '设置条件值 2 复合框默认值
    Call 清除显示信息                                         '清除 ListView 控件数据显示
End Sub

Private Sub 运算符_Change()
    If 运算符.Value <> "between" Then
        '当运算符设置值不为 between 时，隐藏部分控件
        Label_and.Visible = False                            '隐藏 And 标签
        Label_Value2.Visible = False                         '隐藏条件值 2 标签
        条件值 2.Visible = False                             '隐藏条件值 2 复合框
        条件值 1.Width = 179                                 '设置条件值 1 复合框宽度
   Else
        '当运算符设置值为 between 时，将隐藏控件显示出来
        Label_and.Visible = True                             '显示 And 标签
        Label_Value2.Visible = True                          '显示条件值 2 标签
        条件值 2.Visible = True                              '显示条件值 2 复合框
        条件值 1.Width = 79                                  '设置条件值 1 复合框宽度
    End If
End Sub
```

11.10.4　重设条件按钮代码设计

重设条件将所有设置条件的复合框控件设置为默认项目，并且清除 ListView 控件的数据显示。单击该按钮后，用户可以对查询条件重新进行设置。以下是该按钮单击事件过程的代码解释：

```
Private Sub 重设条件_Click()
    信息种类.ListIndex = 0                                   '设置信息种类显示值
    查询项目.ListIndex = 0                                   '设置查询项目显示值
    运算符.ListIndex = 0                                     '设置运算符显示值
    条件值 1.ListIndex = 0                                   '设置条件值 1 显示值
    条件值 2.ListIndex = 0                                   '设置条件值 2 显示值
    Call 清除显示信息                                         '清除 ListView 控件显示
End Sub
```

11.10.5　开始查询按钮代码设计

用户在合同信息查询与导出窗口中单击【开始查询】按钮后，程序将按照用户设置的查

询条件查询数据库，然后将查询结果显示在ListView 控件中。该按钮的单击事件过程流程图如图 11-51 所示。

从图中可以看出【开始查询】按钮完成的任务数量很少，但是其代码并不简单。以下是该按钮单击事件的代码解释：

获取查询字符串
生成查询记录集
显示查询记录数据

图 11-51 【开始查询】按钮单击事件过程流程图

```
Private Sub 开始查询_Click()
    Dim SQL As String
    Dim Condition As String, Con0 As String, Con1 As String, Con2 As String
    '设置查询条件
    Con0 = " where "
    If 查询项目.Value = "签订日期" Or 查询项目.Value = "合同起始日" _
        Or 查询项目.Value = "合同终止日" Or 查询项目.Value = "收费日期" Then
        Con1 = "#" & Format(条件值 1.Value, "yyyy-mm-dd") & "#"
        Con2 = "#" & Format(条件值 2.Value, "yyyy-mm-dd") & "#"
    ElseIf 查询项目.Value = "合同金额" Or 查询项目.Value = "收费金额" Then
        Con1 = Val(条件值 1.Value)
        Con2 = Val(条件值 2.Value)
    Else
        Con1 = "'" & 条件值 1.Value & "'"
        Con2 = "'" & 条件值 2.Value & "'"
    End If
    Condition = " where " & 查询项目.Value
    If 运算符.Value = "between" Then
        Condition = Condition & " between " & Con1 & " and " & Con2
    ElseIf 运算符.Value = "like" Then
        Condition = Condition & " like '%" & 条件值 1.Value & "%'"
    Else
        Condition = Condition & 运算符.Value & Con1
    End If
    '设置 SQL 语句
    SQL = "select * from " & 信息种类.Value & Condition
    '开始查询
    Set rs = New ADODB.Recordset
    rs.Open SQL, cnn, adOpenKeyset, adLockOptimistic
    If rs.BOF And rs.EOF Then
        MsgBox "没有查询到结果！", vbCritical, "查询结果"
        Exit Sub
    End If
    '将查询结果显示在 ListView 控件中
    With ListView1
        '设置 ListView1 的标题、显示类型、整行选择和网格线属性
        .ColumnHeaders.Clear
        .ListItems.Clear
        .View = lvwReport
        .FullRowSelect = True
        .Gridlines = True
```

```
            '为 ListView1 设置标题
            For i = 0 To rs.Fields.Count -1
                .ColumnHeaders.Add , , rs.Fields(i).Name
            Next i
            '为 ListView1 设置各行数据
            .ListItems.Clear
            For i = 1 To rs.RecordCount
                .ListItems.Add , , rs.Fields(0).Value
                For j = 1 To rs.Fields.Count -1
                    .ListItems(i).SubItems(j) = rs.Fields(j).Value
                Next j
                rs.MoveNext
            Next i
            rs.MoveFirst
        End With
End Sub
```

11.10.6　数据导出按钮代码设计

【数据导出】按钮用于将用户查询到的记录数据导出到单个的 Excel 文件中。程序首先生成一个新的工作簿，然后将记录数据的表头写入工作表中，并设置格式，最后将记录数据依次写入表中。该过程的流程图如图 11-52 所示。

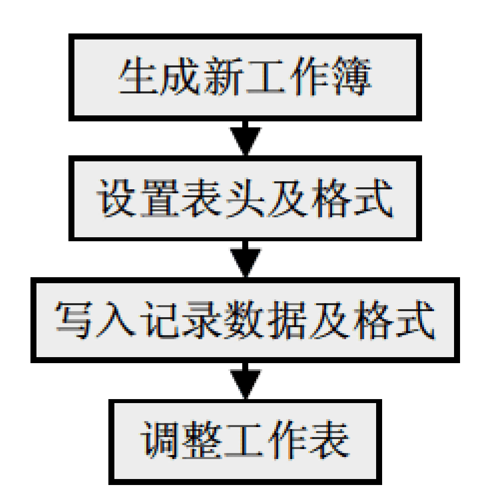


图 11-52　【数据导出】按钮单击事件过程流程图

以下是该按钮单击事件过程的代码解释：

```
Private Sub 数据导出_Click()
        Dim wb As Workbook
        Dim ws As Worksheet
        Dim i As Integer, j As Integer
        Set wb = Workbooks.Add                              '添加新工作簿
        Set ws = wb.ActiveSheet                             '指定写入数据的工作表
        With ws
            '设置表头
            For i = 0 To rs.Fields.Count -1
                .Cells(1, i + 1) = rs.Fields(i).Name        '将表头写入标题行对应单元格中
            Next
            '设置表头格式
            With .Range(Cells(1, 1), Cells(1, rs.Fields.Count))
                .Font.Bold = True                           '字体加粗
                .HorizontalAlignment = xlCenter             '水平居中对齐
            End With
            '写入记录数据
            For i = 1 To rs.RecordCount
                For j = 0 To rs.Fields.Count -1
                    .Cells(i + 1, j + 1) = rs.Fields(j)     '将记录字段数据写入对应单元格中
                    If rs.Fields(j).Type = adDate Then
```

```
                .Cells(i + 1, j + 1).NumberFormat = "yyyy-mm-dd"       '设置日期型记录数据的显示格式
            End If
            If rs.Fields(j).Type = adCurrency Then
                .Cells(i + 1, j + 1).NumberFormat = "#,##0.00"         '设置货币型记录数据的显示格式
            End If
        Next
        rs.MoveNext                                    '将记录指针移动到下一条
    Next i
    .Columns.AutoFit                                   '列自动对齐
  End With
  Set ws = Nothing                                     '清除工作表对象临时变量
  Set wb = Nothing                                     '清除工作簿对象临时变量
End Sub
```

11.10.7　清除显示信息过程代码设计

清除显示信息过程用于将 ListView 的显示数据进行重置。这些数据包括了该控件的标题项目以及数据显示项目。以下是该自定义过程的代码解释：

```
Public Sub 清除显示信息()
  With ListView1
    .ColumnHeaders.Clear                               '清除控件所有标题
    .ListItems.Clear                                   '清除控件所有数据项目
    .View = lvwReport                                  '定义控件显示模式
    .FullRowSelect = True                              '允许整行选择
    .Gridlines = True                                  '显示网格线
  End With
End Sub
```

11.11　系 统 测 试

当该文件被打开时，系统会自动弹出登录窗口。当然可以不输入任何登录信息，但是用户将不能完成合同基本信息管理、合同收费信息管理和合同信息查询与导出 3 部分的工作。本系统的功能基本上是通过窗口实现的，系统共包含了 6 个窗口，下面将分 5 个部分分别测试这 5 个窗口的功能。

11.11.1　登录窗口测试

本系统只有登录后才能打开相应的合同管理工作簿，否则无法打开 3 个合同管理窗口。要登录工作簿，操作步骤如下：

（1）找到该工作簿文件或在系统的首页单击【登录系统】按钮后，将弹出【用户登录】

对话框（如图 11-53 所示）。

（2）系统只建立一个用户与密码。第一次运行时，在【用户名】文本框中输入“admin”，在【密码】文本框中输入“123456”，然后单击【进入系统】按钮。

当用户输入了错误的用户名或者错误的密码，系统将自动退出。在退出之前，系统会弹出一个错误提示框告知用户用户名和密码错误（如图 11-54 所示）。

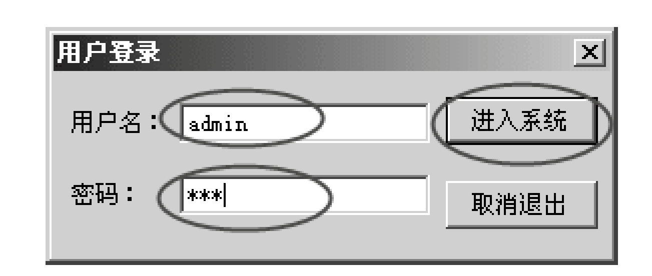


图 11-53　【用户登录】对话框



图 11-54　错误提示框

11.11.2　修改用户名窗口测试

用户在登录与未登录到系统的状态下，都可以完成用户名的修改操作。要完成修改用户名操作，用户需要在【修改用户名】对话框中输入 3 条信息，分别是原用户名、新用户名和用户密码。修改用户名的操作过程如下：

（1）在首页中单击【修改用户名】按钮，打开【修改用户名】对话框。

（2）在【原用户名】、【新用户名】和【密码】文本框中分别输入相应信息，然后单击【确定】按钮（如图 11-55 所示）。

（3）用户输入错误密码后，系统会弹出一提示框（如图 11-56 所示）。单击【确定】按钮后，【修改用户名】对话框的【密码】文本框数据被清除。此时的鼠标焦点也将处于该文本框中（如图 11-57 所示）。

（4）当用户输入密码正确时，系统将保存该用户名修改结果，并提示用户修改成功（如图 11-58 所示）。



图 11-55　【修改用户名】对话框



图 11-56　修改用户名密码错误



图 11-57　重设修改用户名密码

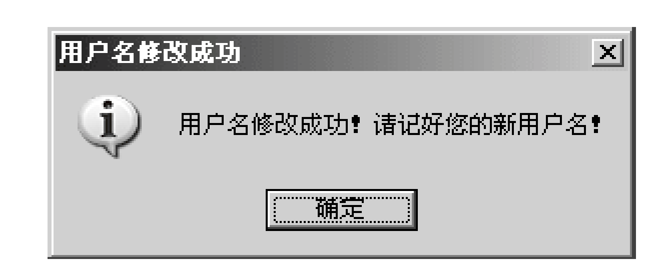


图 11-58　修改用户名成功

11.11.3　修改用户密码窗口测试

用户在登录与未登录到系统的状态下，都可以完成用户密码的修改操作。要完成修改用户密码操作，用户需要在【修改密码】对话框中输入 3 条信息，分别是用户名、原密码和新密码。修改用户密码操作的过程如下：

（1）在首页单击【修改用户名】按钮，打开【修改密码】对话框。

（2）在【用户名】、【原密码】和【新密码】文本框中分别输入相应信息，然后单击【确定】按钮（如图 11-59 所示）。

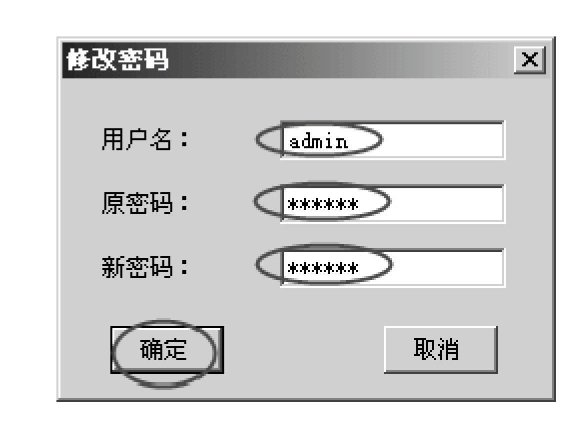


图 11-59　【修改密码】对话框

11.11.4　合同信息管理窗口测试

在本节的测试过程中，不涉及窗口中的浏览按钮。这里讲述的主要是该窗口中的【新合同】、【添加】、【修改】、【删除】以及【查询】按钮的操作。测试该窗口的过程如下：

（1）在首页单击【合同基本信息管理】按钮，弹出【合同基本信息管理】对话框，然后在该对话框中单击【新合同】按钮，此时效果如图 11-60 所示。

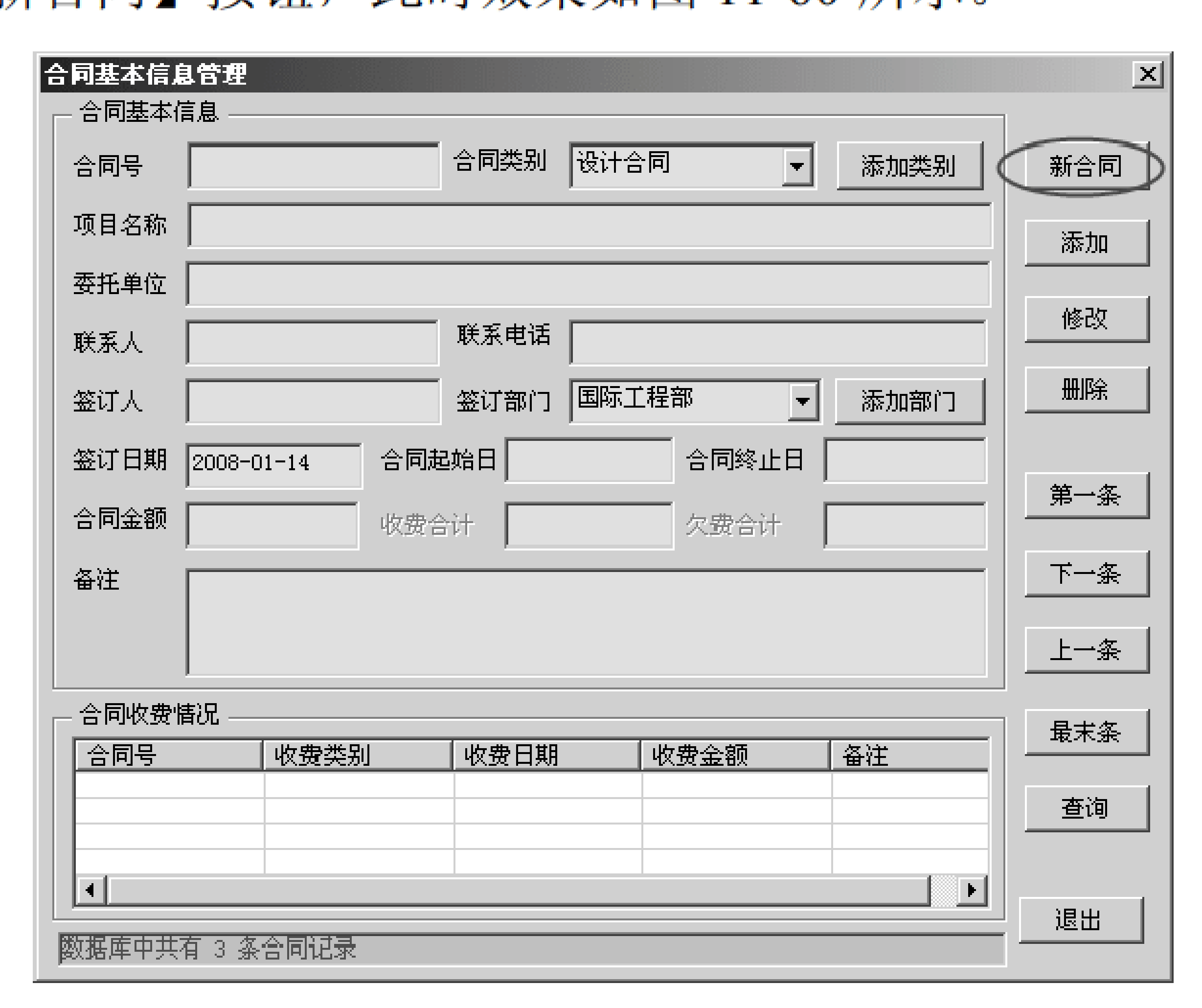


图 11-60　【合同基本信息管理】对话框

（2）在合同基本信息框架中输入合同必需信息（如图 11-61 所示），单击【添加】按钮，在随后弹出的【添加记录】询问框（如图 11-62 所示）中单击【是】按钮，即可将新合同记录添加到数据库中。添加完成后，窗口中显示的该记录并不是新添加的记录。要查看该记录，可以单击【最末条】按钮，也可以单击【查询】按钮。



图 11-61　添加新合同信息记录



图 11-62　确认添加新合同



图 11-63　输入查询合同号

（3）单击【查询】按钮，在随后弹出的合同号输入框中输入“LWH0003”，然后单击【确定】按钮（如图 11-63 所示）。此时窗口中将会立即显示该新添加合同的合同基本信息。

（4）在该新添加的合同中，对联系电话做出修改。这里将联系电话修改为“0311-66666666”，然后单击【修改】按钮。修改成功后弹出一提示框，告知用户合同记录修改成功，确认即可。

（5）单击【浏览】按钮，确保当前窗口显示的合同记录即为前面修改的记录，然后单击【删除】按钮，此时该记录将从数据库中被删除。随后会弹出一提示框告知用户该记录已经被删除，单击【确定】按钮即可。

11.11.5　合同收费信息管理窗口测试

合同收费信息管理窗口中很多按钮和合同基本信息管理窗口按钮类似，并且操作上也相差不大。这里只对修改按钮与添加新收费类别的操作步骤加以测试，其他按钮的功能请读者自己测试。该演示的测试是将 CADT00001 号合同的第一条收费信息中的收费类别为“预付款”。笔者先前将数据库中收费类别的“预付款”已经删除，因而这里还需要新添加“预付款”收费类别。测试过程如下：

（1）在首页单击【合同收费信息管理】按钮，弹出【合同收费信息管理】对话框（如图 11-64 所示）。

（2）在【收费类别】复合框中输入新收费类别【预付款】，然后单击【添加类别】按钮（如图 11-65 所示）。随后会弹出一提示添加成功的提示框，单击【确定】按钮即可（如图 11-66 所示）。当用户再次选择【收费类别】复合框右侧的下拉列表框时，此时将会显示预付款的选项供用户选择（如图 11-67 所示）。

（3）在【合同收费信息管理】窗口单击【修改】按钮，程序将自动保存用户在前面做出的收费类别修改操作。



图 11-64 【合同收费信息管理】对话框



图 11-65 修改收费类别



图 11-66 添加类别成功

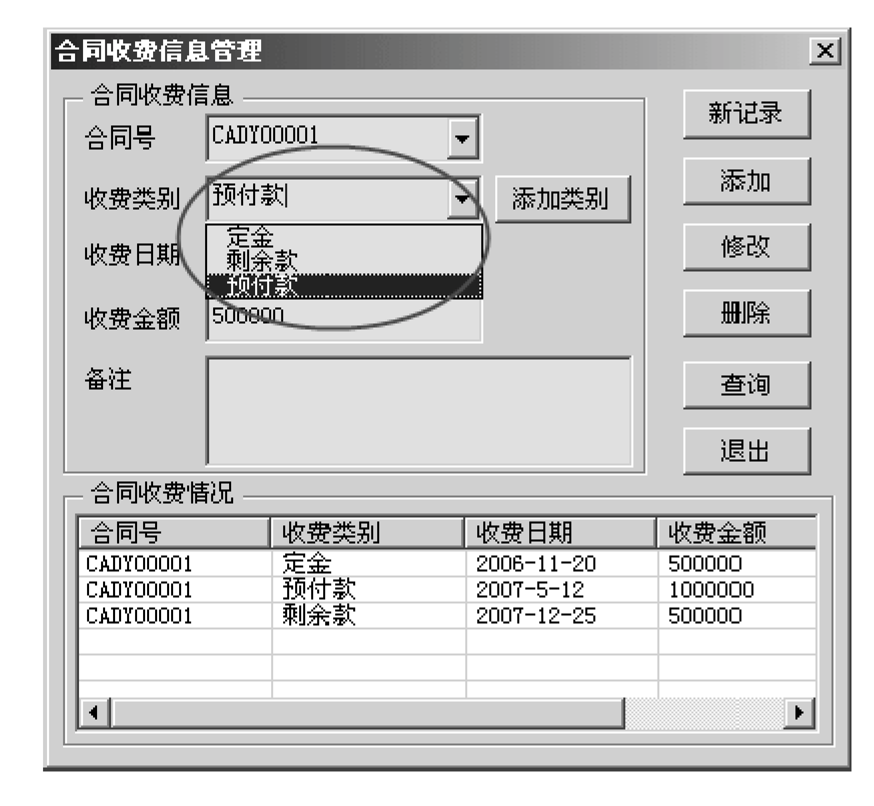


图 11-67 新收费类别被添加

11.11.6 合同信息查询与导出窗口测试

在【合同信息查询与导出】窗口中可以完成合同基本信息与收费信息的查询与导出工作。这里只介绍合同基本信息的查询与导出测试操作。

（1）在首页单击【合同信息查询与导出】按钮，打开【合同信息查询与导出】窗口（如图 11-68 所示）。

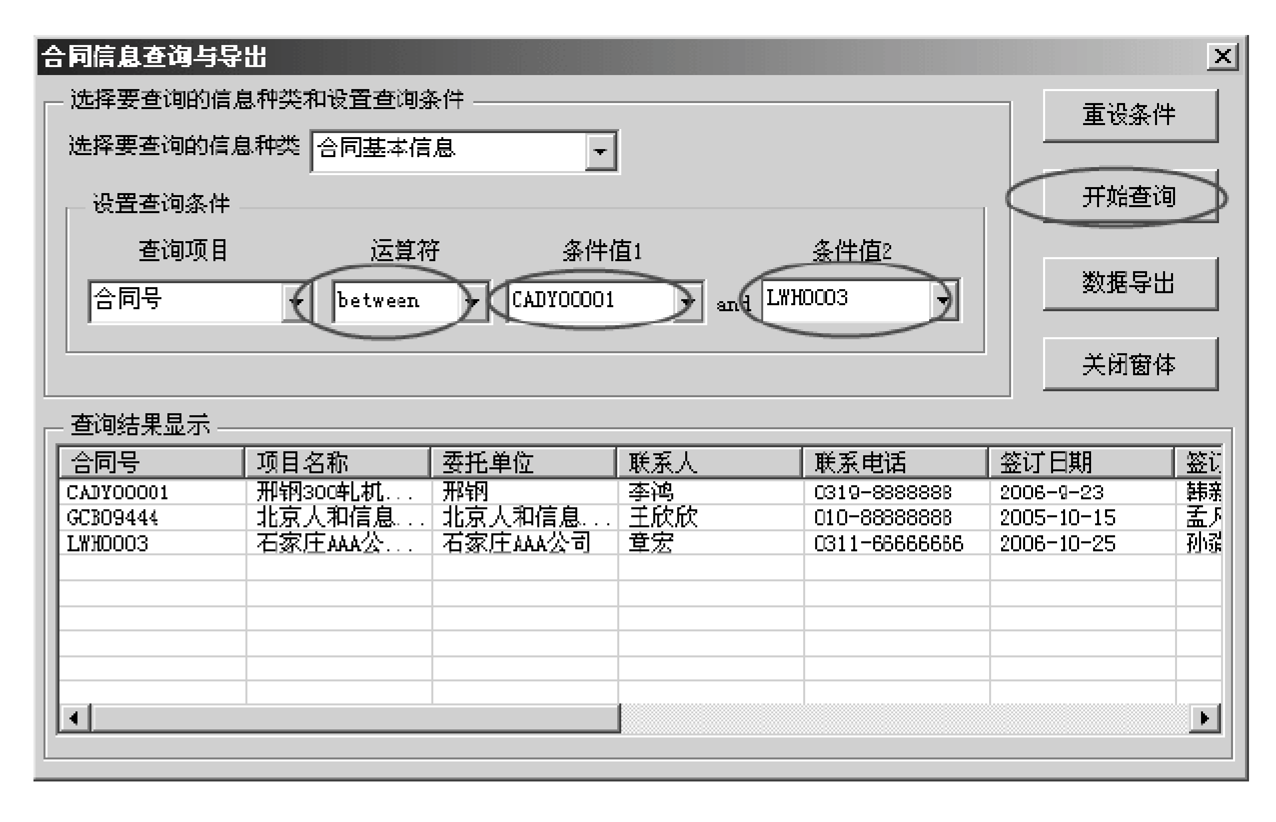
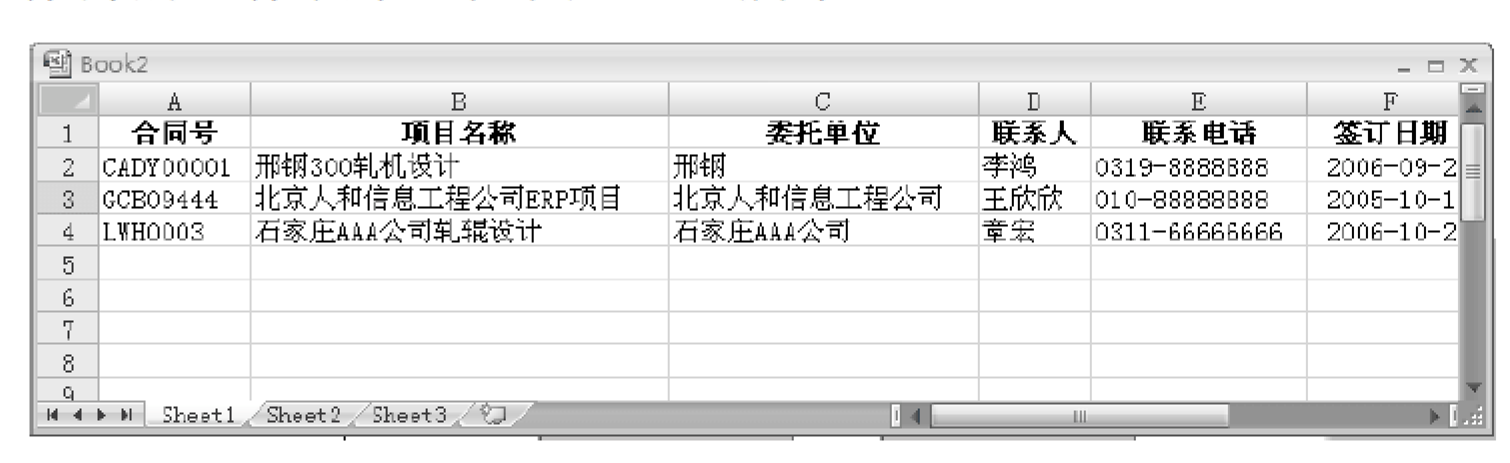


图 11-68 查询合同信息

（2）在选择查询的信息种类中选择【合同基本信息】项目，在【查询项目】复合框中选择【合同号】项目，在【运算符】复合框中选择 between。随后在【条件值 1】和【条件值 2】复合框中分别设置合同号为“CADTY00001”和“LWH0003”，最后单击【开始查询】按钮。此时所有的查询结果被显示在窗口下方的 ListView 控件中。

（3）在窗口中单击【数据导出】按钮，程序将新建一个工作簿，并将查询所得记录集的数据复制到该工作簿的工作表中（如图 11-69 所示）。


Book2

	A	B	C	D	E	F
1	合同号	项目名称	委托单位	联系人	联系电话	签订日期
2	CADY00001	邢钢300轧机设计	邢钢	李鸿	0319-8888888	2006-09-2
3	GCB09444	北京人和信息工程公司ERP项目	北京人和信息工程公司	王欣欣	010-88888888	2005-10-1
4	LWH0003	石家庄AAA公司轧辊设计	石家庄AAA公司	章宏	0311-66666666	2006-10-2
5						
6						
7						
8						
9						

Sheet1 Sheet2 Sheet3

图 11-69 导出查询结果

第 12 章　拆分与备份工作簿系统

拆分与备份工作簿系统是一个能运行在 Excel 2007 下的工作簿工具系统，通过本系统的拆分模块可以将包含多个工作表的工作簿按照用户的组别设置保存到新的工作簿中。备份模块可以将原来分散到各个工作簿中的部分工作表合并到一个新的工作簿中。本工具系统是笔者开发的办公工具作品之一，系统采用加载宏的形式。用户可以将该加载宏安装到自己的 Excel 2007 中，以后每次打开 Excel 2007 都会在加载项菜单中找到该工具的开启菜单。

12.1　系 统 概 述

从该系统的名称上可以看到，本系统只包含了拆分工作簿和备份工作簿两个功能。拆分工作簿是将某工作簿中的工作表分别保存到各个工作簿中，原来的工作簿并不发生变更。备份工作簿是将原来分属不同工作簿的工作表合并到同一个工作簿。原来笔者命名该系统时是以拆分和合并工作簿为名称的。

该系统的架构十分简单。在拆分工作簿时，首先获取源工作簿，然后在设置需要拆分出来的工作表，最后保存拆分工作簿。备份工作簿时，首先选择备份源工作簿，然后选择需备份的工作表，最后保存备份工作簿。该系统的框架结构图如图 12-1 所示。

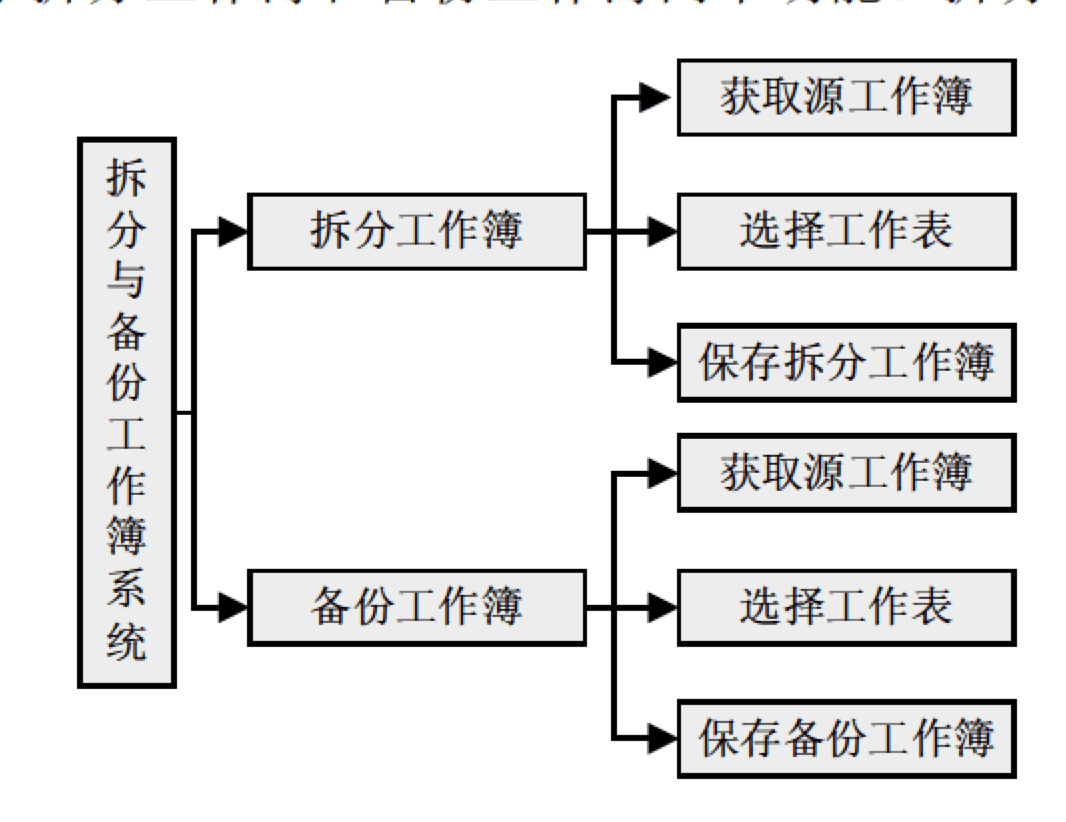


图 12-1　拆分与备份工作簿系统框架结构图

12.1.1　设计思路

拆分与备份工作簿工具系统一共完成两个任务：拆分单个工作簿的工作表到新工作簿和合并各个工作簿的工作表到新工作簿中。前一个功能由一个窗体完成，后一个功能由 4 个窗体完成。以下是这些窗体的功能介绍：

- Frm 拆分工作簿：在该窗体中，用户可以设置需要拆分的工作簿的路径，然后对各个表进行分组。分组的名称将作为其新工作簿的名称。用户还可以在该窗体下对分组内所包含的工作表进行调整，最后通过拆分按钮实现拆分工作。
- Frm 选择工作簿：该窗体主要是选择需要合并工作表的工作簿，在该窗体下用户可以一次选择多个工作簿的名称。选择完成之后，用户还可以再次确认最终真正需要的



工作簿。只有被选中的工作簿才会被系统记录到下一步的表调整操作中。

- Frm 选择工作表：该窗体中，用户可以确定所有需要合并的工作表。这些工作表属于先前被选中的工作簿。
- Frm 保存位置：该窗体中主要用于完成设置保存文件位置并开始合并备份工作。它还可以显示出所有已经确定合并的工作表及所属工作簿的位置信息，以便于用户回到上一步做出调整。
- Frm 提示信息：该窗体用于显示一些提示信息。

系统只包含了一个公共模块。公共过程与函数不多，因而将所有的公共变量、过程和函数都保存在同一个模块中。程序中数据的保存是通过数据库实现的。但是其中关于中英文显示界面的实现部分的数据保存在工作表中，关于这部分内容，请参见后续详细介绍。

12.1.2　知识点一：在 Excel 2007 中装载加载宏

在 Excel 2007 中，标准加载宏文件的后缀名称为.xlam。这些加载宏文件可以随同 Excel 2007 一同装载。当确认装载这些文件后，每次开启 Excel 2007 程序，该加载宏都会自动开启。要让加载宏文件能随 Excel 2007 程序一同加载，可按照以下步骤设置：

（1）单击 Excel 2007 左上方的 Office 菜单按钮图标，在随后弹出的菜单中单击【Excel 选项】按钮，如图 12-2 所示。



图 12-2　打开【Excel 选项】

（2）在【Excel 选项】窗口中（如图 12-3 所示），选择【加载宏】项目。在其右侧底部找到【管理】复合框，选择【Excel 加载项】选项，然后单击其右侧的【转到】按钮，打开【加载宏】管理窗口。

（3）在【加载宏】管理窗口中（如图 12-4 所示），单击【浏览】按钮。找到需要自动加载的加载宏文件并选择该文件，随后该文件的文件名将被添加到【可用加载宏】列表框中。

在该列表框中选择已经添加到管理器中的加载宏文件并单击【确定】按钮即可。



图 12-3　加载项设置

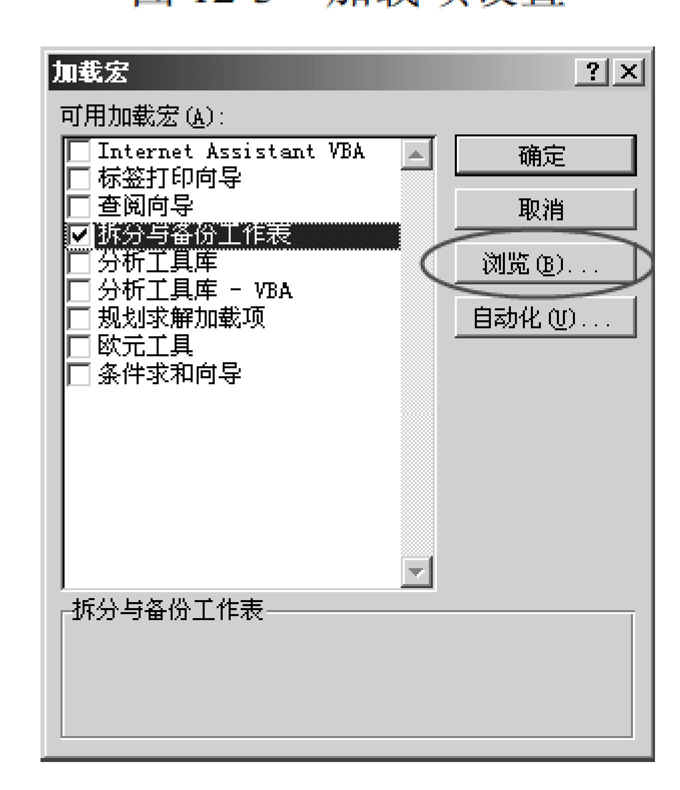


图 12-4　【加载宏】管理窗口

12.1.3　知识点二：使用 ADOX 库

在本实例中，需要快速确定工作簿中包含的工作表，并且要将这些工作表的名称体现在窗口列表控件中。要获取这些信息，通过 ADO 对象比较困难。这里使用了 ADO 的一个扩展库 ADOX，通过使用该对象库的 Catalog 和 Table 对象可以迅速确定工作簿中包含的工作表及名称。调用该对象库的方法如下：

在 VBE 开发环境中依次选择【工具】|【引用】命令（如图 12-5 所示），打开【引用】对话框，在该对话框中找到 Microrsoft ADO Ext.2.8 For DLL and Security 的引用（如图 12-6 所

示）。选中该复选框后，单击【确定】按钮即可。关于该对象库的使用方法，请见后续章节的介绍。



图 12-5　引用菜单

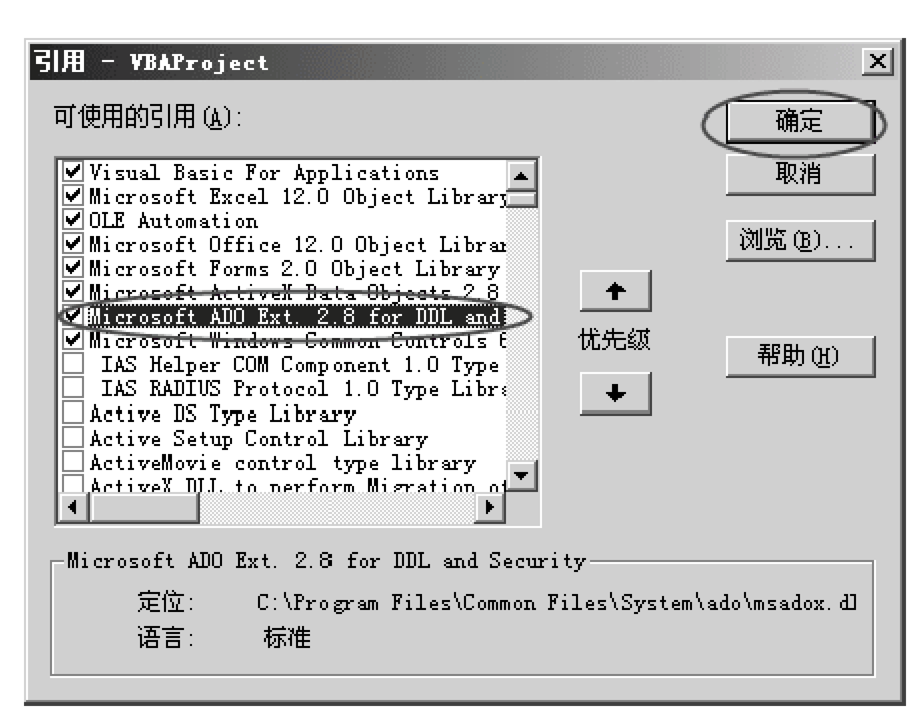


图 12-6　引用 ADOX 对象库

12.2　数据库表设计

系统中几乎大部分的临时数据都保存在了数据库表中。该数据库通过 Access 2007 建立，该文件被保存为“临时数据.accdb”，读者可以在随书光盘中找到该文件。在该数据库中包含了 4 个数据表，分别是拆分表、拆分表组别、工作表和工作簿。这 4 个表的作用分别如下：

- 拆分表：该表用于记录需拆分的工作簿中所有表的分组情况。当执行拆分时，该表保存的信息决定了拆分的方式，即工作簿中各个工作表的归属情况。
- 拆分表组别：该表存储非重复的工作表分组信息。这些分组名称最终会成为各个拆分工作簿的名称（不包含工作簿的后缀，后缀取原文件的后缀）。而各个工作表在拆分时，将按照拆分表信息依次归入各个新工作簿中。
- 工作簿：该表保存的是用户在备份工作簿窗口中打开的所有工作簿的路径与名称信息等。该表的信息包含了用户选中和未选中的所有工作簿资料。
- 工作表：该表包含了所有用户选定合并备份的工作表的表名和所属工作簿路径信息。

表 12-1~表 12-4 以表格的形式说明这些表的字段结构设计情况。此处不再具体说明在 Access 2007 中建立这些表的过程。如果读者对此不很明了，可以参见 9.2 节具体内容。

表 12-1　拆分表字段设计

字 段 名 称	数 据 类 型	字 段 长 度	是否允许为空
工作表名	文本	50	是
组名	文本	50	是

表 12-2　拆分表组别字段设计

字段名称	数据类型	字段长度	是否允许为空
组名	文本	50	是

表 12-3　工作表字段设计

字 段 名 称	数 据 类 型	字 段 长 度	是否允许为空
工作簿路径	文本	255	是
工作表名	文本	40	是

表 12-4　工作簿字段设计

字 段 名 称	数 据 类 型	字 段 长 度	是否允许为空
工作簿路径	文本	255	是
工作簿名	文本	40	是
是否选定	是/否	2	否

12.3　工作簿与公共模块代码设计

工作簿对象和公共模块中包含的代码大多都是完成系统公用变量定义或公共设置。了解该部分的代码设计有利于搞清楚系统整体设计与程序运行的流程。该节介绍的部分过程和函数将被其他窗体或过程调用，在阅读后续章节时，读者可能需要重新回到本节参阅公共函数或过程的代码。因此，这里将这部分代码的说明放在本章的前端。

工作簿对象的事件代码十分简单，但公共模块代码较长。后续小节将把公共模块的所有过程和函数单独列出加以介绍。

12.3.1　工作簿对象代码设计

工作簿对象中的代码只包含了两个事件代码。这两个事件的代码并不长，这里将不对这些代码分开介绍。其功能描述如下：

- 工作簿开启事件：工作簿开启时，需要完成的工作包括数据库链接的设置、系统菜单的设计。工作簿一打开，程序将立即链接到“临时数据.accdb”数据库文件，该链接被保存在公共变量“cnn 临时数据簿”中。
- 工作簿关闭事件：和工作簿开启事件相对应。关闭工作簿时，需要清除系统建立的菜单系统并且断开数据库链接。

以下是这两个事件过程的代码解释：

```
Private Sub Workbook_Open()
Dim objMenu As CommandBarPopup, itemMenu As Object
'打开数据库链接
cnn 临时数据簿.Open GetConnString(ThisWorkbook.Path & "\临时数据.accdb")
cnn 临时数据簿.Execute "delete * from  工作簿"                '删除工作簿表中保存的数据
cnn 临时数据簿.Execute "delete * from  工作表"                '删除工作表中保存的数据
'为系统建立菜单
Set objMenu = Application.CommandBars(1).Controls.Add(Type:=msoControlPopup, _
                before:=10, temporary:=True)                   '添加一级菜单对象
```

```
objMenu.Caption = "工作簿拆分与备份"                        '设置一级菜单 Caption 属性
'添加拆分工作簿菜单
Set itemMenu = objMenu.Controls.Add(Type:=msoControlButton)        '添加第一个二级菜单按钮
itemMenu.OnAction = "显示拆分窗口"                          '设置第一个二级菜单按钮执行过程
itemMenu.Caption = "拆分工作簿"                             '设置第一个二级菜单 Caption 属性
'添加备份工作簿菜单
Set itemMenu = objMenu.Controls.Add(Type:=msoControlButton)        '添加第二个二级菜单按钮
itemMenu.OnAction = "显示备份窗口"                          '设置第二个二级菜单按钮执行过程
itemMenu.Caption = "备份工作簿"                             '设置第二个二级菜单 Caption 属性
Set objMenu = Nothing                                       '清除 objMenu 变量
Set itemMenu = Nothing                                      '清除 itemMenu 变量
End Sub

Private Sub Workbook_BeforeClose(Cancel As Boolean)
On Error Resume Next                                        '发生错误时继续执行下一条语句
Application.CommandBars(1).Controls("工作簿拆分与备份").Delete        '删除
On Error GoTo 0                                             '恢复默认错误处理机制
Set cnn 临时数据簿 = Nothing
End Sub
```

代码说明：

在工作簿打开事件过程中，打开数据库链接时，使用了一个 GetConnString 自定义函数。该函数接受一个数据库文件的绝对地址，然后返回一个完整的链接到该数据库文件的链接字符串。

12.3.2　公共变量与菜单按钮代码设计

本小节介绍的是公共模块中所有公共变量的实际意义和菜单按钮的过程代码。这部分内容是公共模块中较为简单的部分，代码也比较少，因而归纳为一类加以介绍。以下是该部分的代码解释：

```
Public languageset As Boolean                               '设定语言种类
'获取在选择工作簿窗口中打开工作簿的路径字符串，由于允许选择多个文件，因此该变量可能是一个数
 组，这里设定该变量的数据类型为 Variant
Public arr 选择工作簿 As Variant
'获取需拆分工作簿的路径，由于设定了不允许选择多个文件，所以固定该变量的参数为字符串
Public str 拆分工作簿 As String
Public cnn 主工作簿 As New ADODB.Connection                 '到本工作簿的链接对象
cnn 临时数据簿 As New ADODB.Connection                      '到数据库文件的链接对象
Public rs As New ADODB.Recordset                            '数据库记录集对象

Public Sub 显示拆分窗口()
frm 拆分工作簿.Show                                         '显示拆分工作簿窗口
End Sub

Public Sub 显示备份窗口()
frm 选择工作簿.Show                                         '显示选择工作簿窗口
End Sub
```

12.3.3　刷新窗体语言显示过程代码设计

刷新窗体语言显示过程是一个通用的窗体语言显示刷新过程。当语言显示设置发生改变时，对于所有的控件而言，需要修改的只是控件的 Caption 属性。在本加载宏工作簿中存在一个唯一的表，该表保存了语言显示的数据资料。该表包含 3 个字段信息，分别是控件的名称（Name）、语言设置为英文时应该显示的字符（EnglishString）、语言设置为中文时应该显示的字符（ChineseString）。

当进入刷新过程后，程序接受了一个窗口对象参数。再根据语言设置获取所有记录的两个字段数据，一个是控件名称，一个是对应语言设置的字段。然后程序根据指定的控件名，在记录集中找到对应控件当前语言设置下显示的字符串。对于窗体和窗体中的控件操作都是一样，不同的是对于窗体中的控件可以使用循环来实现逐个修改显示信息。

获取对应语言设置的记录集 → 记录集中查找当前窗体的标题显示字段 → 设置窗体的标题 → 循环窗体控件，设置控件标题

图 12-7　刷新窗体语言显示过程流程图

如图 12-7 所示的是该过程的流程图。

以下是该过程的代码解释：

```
Public Sub 刷新窗体语言显示(formObject As Object)
On Error Resume Next                                '代码发生错误时继续执行下一条语句
rs.Close                                            '关闭 rs 记录集对象，以免发生重用错误
On Error GoTo 0                                     '恢复默认错误处理机制
'根据语言设置变量，获取指定语言设置下对应记录集
If languageset Then
     '获取控件名称和英文显示字段数据
     rs.Open "select Name,EnglishString from [语言设置$]", cnn 主工作簿, adOpenKeyset,
adLockOptimistic
Else
     '获取控件名称和中文显示字段数据
     rs.Open "select Name,ChineseString from [语言设置$]", cnn 主工作簿, adOpenKeyset,
adLockOptimistic
End If
On Error Resume Next
rs.Find "Name='" & formObject.Name & "'"            '在记录集中找到该窗体名
formObject.Caption = rs.Fields(1)                   '指定窗体的标题
rs.MoveFirst                                        '将记录集指针归位
'循环所有窗口中的控件，为对应控件指定标题
For Each i In formObject.Controls
     rs.Find "Name='" & i.Name & "'"                '在记录集中找到该控件名
     i.Caption = rs.Fields(1)                       '指定控件的标题
     rs.MoveFirst                                   '将记录集指针归位
Next
On Error GoTo 0
rs.Close                                            '关闭 rs 记录集对象
```

```
Set rs = Nothing
End Sub
```

12.3.4　刷新工作簿列表过程代码设计

刷新工作簿列表过程用于刷新选择工作簿窗口中显示选择工作簿的 ListView 控件项目的过程。在选择工作簿窗口第一次被加载或者再次被显示出来时都会被执行一次，以便 ListView 控件能正确反映用户做出的操作。

在数据库文件中的工作簿表中，存储的就是有关工作簿选择的相关信息资料。要刷新工作簿列表，只需要将该工作簿中的数据读取出来，然后根据这些数据进行设置即可。

程序首先从数据库中获取了工作簿表的所有数据信息，然后清空了 ListView 控件的显示项目。当记录集没有记录时，说明没有工作簿被选择，直接退出过程即可。当存在记录时，程序将为 ListView 控件添加新项目，然后依次根据记录字段信息确定控件对应各列的显示数据。在这中间程序还确定了【下一步】按钮的可用状态。在程序中，如果没有一个选择工作簿被选中，【下一步】按钮将不可用。如图 12-8 所示的是该过程的流程图。

图 12-8　刷新工作簿列表过程流程图

以下是该过程的代码解释：

```
Public Sub 刷新工作簿列表()
Dim strSQL As String
'打开到数据库工作簿表的记录集
rs.Open "select * from [工作簿]", cnn 临时数据簿, adOpenKeyset, adLockOptimistic
frm 选择工作簿.List 工作簿.ListItems.Clear                      '清空 List 工作簿控件项目
If rs.RecordCount = 0 Then Exit Sub                             '记录集为空时直接退出过程
Do Until rs.EOF                                                 '循环到记录集终端
    With frm 选择工作簿.List 工作簿.ListItems.Add(Text:=rs.Fields("工作簿名"))     '添加项目
        .SubItems(1) = rs.Fields("工作簿路径")                   '添加新项目的第一个子项目
        '确定项目是否被勾选
        If rs.Fields("是否选定") = True Then
            .Checked = True                                     '项目被选中
        Else
            .Checked = False                                    '项目不被选中
        End If
        If .Checked Then frm 选择工作簿.btn 下一步.Enabled = True     '确定下一步按钮可用状态
    End With
    rs.MoveNext                                                 '移动记录集指针到下一条
Loop
```

```
rs.Close                                        '关闭记录集
Set rs = Nothing
End Sub
```

12.3.5　保存选择工作簿代码设计

在选择工作簿窗口中，用户需要将包含合并工作表的所有工作簿都一一显示在工作簿列表控件中。用户可以通过【打开】按钮获取对应工作簿文件的详细路径。该获取方式可以一次获取多个工作簿的路径。单击其中的【打开】按钮弹出一个【系统打开】对话框，用户选择文件后，实际上不是真正打开了选择文件，而是程序将这些文件的详细路径记录下来，以便程序实现窗口工作簿列表的刷新操作。

保存选择工作簿过程正是为保存用户每次单击【打开】按钮后的选择文件详细路径而用。这些数据将被保存到数据库的工作簿表工作簿路径字段中，同时还确认了该表的其他字段的数据。

过程的主体是一个 For 循环。该循环的循环变量以打开工作簿的总数量开始，步长为-1，循环到 1 为止。循环体中，程序首先确认打开工作簿是否在列表已经显示。如果已经显示，则进入下一个打开工作簿的检测工作。

在选择工作簿窗口中打开了工作簿后，工作簿列表将把这些工作簿一一记录。但是在进入下一步工作表的选择前，需要在列表中选中至少一个工作簿。保存选择工作簿的过程就是用于将用户选中的工作簿在数据库中加以标记，而没有选中的工作簿则去掉标记，以便选择工作簿窗口再次被打开时，窗口中显示的是用户先前的选择设置。程序将把该工作簿的路径与文件名保存到数据库的工作簿表中，并设置工作簿为选中状态。如图 12-9 所示的是该过程的流程图。

图 12-9　保存选择工作簿过程流程图

以下是该过程的代码解释：

```
Public Sub 保存选择工作簿()
Dim strSQL As String
Dim itemlist As ListItem
On Error Resume Next                        '发生错误时，继续执行下一条语句
For i = UBound(arr 选择工作簿) To 1 Step -1
    strSQL = "select * from [工作簿] where 工作簿路径='" & arr 选择工作簿(i) & "'"   '设置查询字符串
    rs.Close                                '关闭 rs 记录集
    rs.Open strSQL, cnn 临时数据簿           '获取工作簿路径为 arr 选择工作簿(i)的工作簿记录集
    If rs.RecordCount > 0 Then
        GoTo Next_For                       '当记录集有记录时，结束该次循环，继续检测下一个工作簿
    Else
        strSQL = "select * from [工作簿]"    '设置查询字符串
```

```
            With rs
                .Close                          '关闭 rs 记录集
                '获取工作簿表所有记录的记录集
                .Open strSQL, cnn 临时数据簿, adOpenKeyset, adLockOptimistic
                .AddNew                         '添加新记录
                .Fields("工作簿路径") = arr 选择工作簿(i)                '设置工作簿路径字段值
                .Fields("工作簿名") =  获取工作簿名(arr 选择工作簿(i))    '设置工作簿名字段值
                .Fields("是否选定") = False                             '设置是否选定字段值
                .Update                                                 '更新工作簿表
            End With
        End If
Next_For:
Next
Set itemlist = Nothing
rs.Close
Set rs = Nothing
On Error GoTo 0
End Sub
```

12.3.6　保存选择工作簿过程代码设计

保存选择工作簿过程正是修改数据库中对应工作簿的“是否选定”字段，以保证窗口在重显时各个工作簿的选定状态的正确性。在选择工作簿窗口完成了工作簿选择后，还需要再次选中需要备份工作表的工作簿，否则窗口中的【下一步】按钮是不会被激活的。

设置选中工作簿“是否选定”字段，可以选择两个方法：一个是选择工作簿窗口工作簿列表中项目被单击时，检测项目的选定状态，进而修改数据库数据；一个是在单击【下一步】按钮时，将选择工作簿窗口工作簿列表中所有项目的选定状态写入到数据库中。选择工作簿窗口采用了第二种方法，而在后面的选择工作表窗口中采用了第二种方法。读者可以比较两种方法的实现方式。

第一种方法不断地在数据库中打开记录集并频繁修改数据，如果对于数据修改的时效性要求较强，应该采用该方法。而第二种方法只打开数据库一次并且一次性完成修改操作，从系统运行速度方面考虑应该尽量采用第二种方法。

以下是该过程的代码解释：

```
Public Sub 保存选中工作簿()
Dim itemlist As ListItem
'打开到数据库工作簿表的记录集
rs.Open "select * from [工作簿]", cnn 临时数据簿, adOpenDynamic, adLockOptimistic
'逐个检测工作簿列表中项目的选定状态，并修改工作簿表中“是否选定”字段的数据
For Each itemlist In frm 选择工作簿.List 工作簿.ListItems
    If itemlist.Checked Then
        rs.Fields("是否选定") = True            '项目为选定时，修改“是否选定”项为真
    Else
        rs.Fields("是否选定") = False           '项目为未选定时，修改“是否选定”项为假
    End If
```

```
        rs.Update                                '更新记录集
        rs.MoveNext                              '将记录集移动到下一条
Next
rs.Close                                         '关闭记录集
Set rs = Nothing
End Sub
```

12.3.7 合并工作簿过程代码设计

合并工作簿是合并备份工作簿模块的核心程序。该程序根据用户设置的保存文件路径与名称、工作簿中包含的工作表，完成合并备份工作。该过程代码很长，这里将分段介绍该过程合并工作簿功能的实现过程。

程序首先根据用户设置的文件保存信息，检测该位置是否已经存在此文件。当存在时，用户可以设置覆盖和添加。覆盖将删除原文件，添加将打开该文件并把表添加到该工作簿中。不存在时，则可以直接添加新工作簿。对于新添加的工作簿，首先需要清除默认建立的工作表。但是工作簿中至少需要有一个工作表，因此有一个表需要在最后完成所有的表复制操作后才可以删除。该表的名称被保存在变量 DeleteID 中。

删除新工作簿中的默认表后，程序将从数据库中获取需要合并的数据表。程序打开记录集时，按照工作簿路径进行了分组。程序根据分组依次打开需备份工作表的工作簿，从记录集中得到需备份的工作表名。当该工作表名与 DeleteID 相同时，需要对 DeleteID 对应表的表名进行修改，否则复制工作表工作将无法正确完成。

将打开工作簿需要复制的表都复制到结果工作簿中后，程序将保存该工作簿后关闭该工作簿，然后打开下一个需要复制工作表的工作簿，继续复制工作表。将所有工作表复制完成后，程序保存该结果工作簿，然后显示提示保存结束信息并结束过程。

由于该过程的流程比较复杂，为了让读者能更准确了解过程的流程，这里将把整个过程拆分成两个流程图。第一个流程图中最后实现将工作表保存到工作簿的过程没有列出明细。第二个流程图展示的是将工作表保存到工作簿的流程。这两个流程图如图 12-10 和图 12-11 所示。

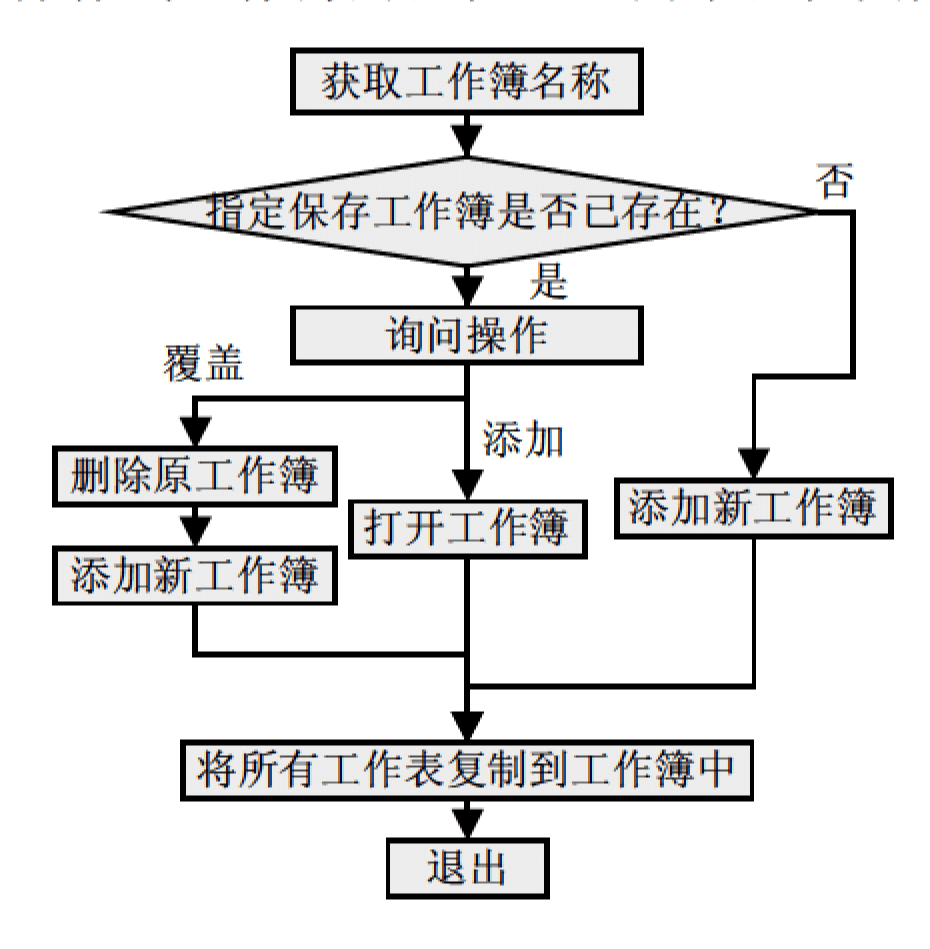


图 12-10 合并工作簿过程流程图

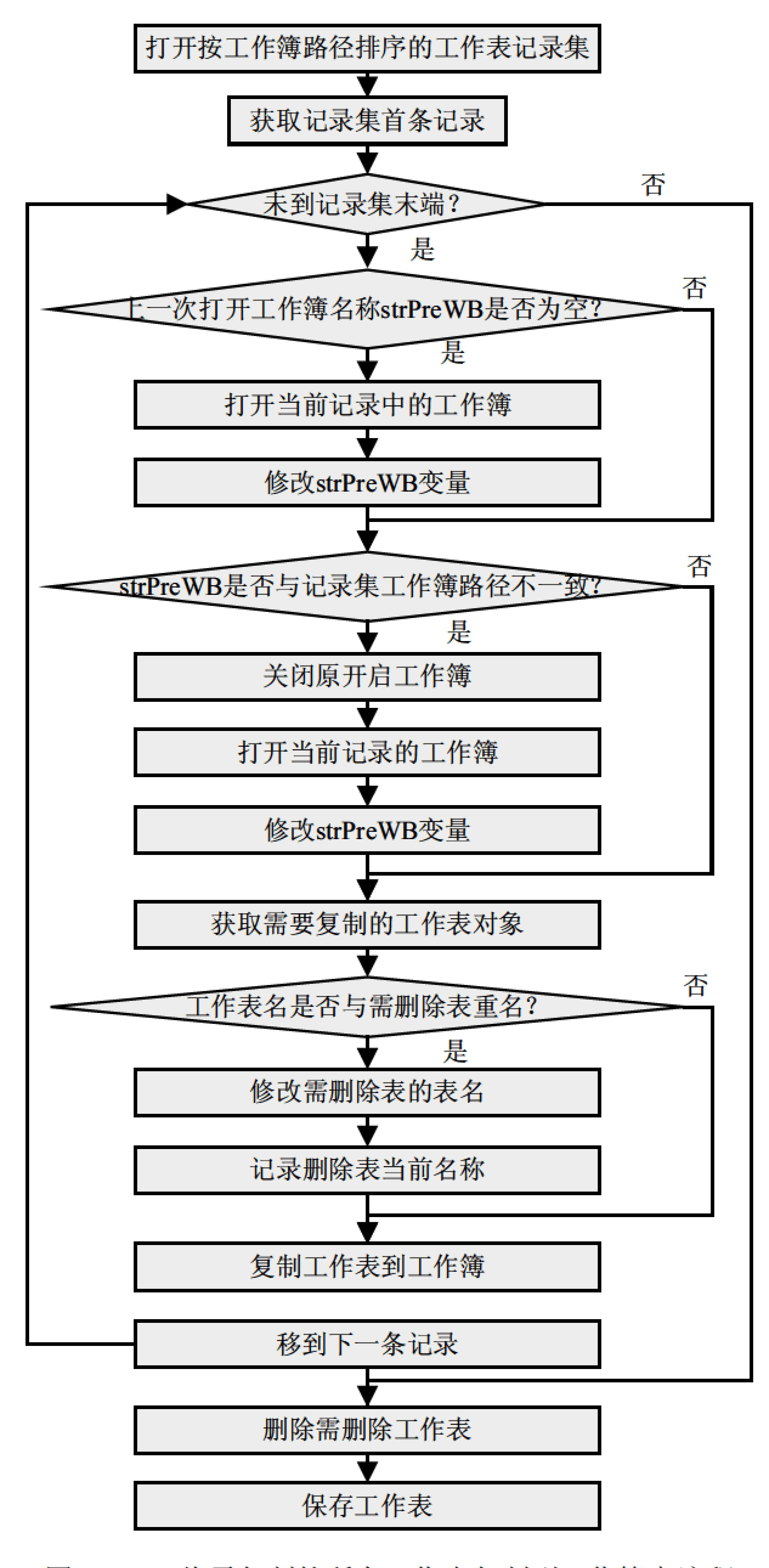


图 12-11　将需复制的所有工作表复制到工作簿中流程

```
Public Sub CombineWorkBook(FileLoc As String)
Dim wk As Workbook, ws As Worksheet
Dim ResultFileLoc As String, resultWK As Workbook, DeleteID As String
Dim strTemp As String, intPos As Integer, strPreWB As String
Application.ScreenUpdating = False
Application.DisplayAlerts = False
'获取工作簿名称，包括后缀名
ResultFileLoc = FileLoc
```

```
strTemp = StrReverse(FileLoc)
intPos = InStr(1, strTemp, Application.PathSeparator)
strTemp = StrReverse(Left(strTemp, intPos -1))
'检测保存工作簿位置是否已经存在该文件，当存在时，提示是否覆盖
'用户可以选择添加，将所有表添加进该工作簿中
'1—表示覆盖文件；2—添加文件；3—取消操作
If Dir(ResultFileLoc) = strTemp Then
    '文件存在时执行以下代码
    Dim frmTemp As New frm 提示信息
    frmTemp.ShowFormTipInf languageset                '显示提示信息窗口
    Select Case BtnClickIndex
        Case Is = 1
            Kill ResultFileLoc                        '删除对应位置中的文件
            Set resultWK = Workbooks.Add              '添加工作簿
            With resultWK
                .SaveAs FileLoc                       '保存工作簿
                Do Until .Sheets.Count = 1            '删除新添工作簿中的表，直到只有一个表
                    .Sheets(1).Delete                 '删除 sheet(1)
                Loop
                DeleteID = .Sheets(1).Name            '记录最后一个表的名称
            End With
        Case Is = 2
            Set resultWK = Workbooks.Open(ResultFileLoc)        '打开工作簿
        Case Is = 3
            GoTo Exit_Sub                                       '终止过程
        Case Else
            GoTo Exit_Sub                                       '终止过程
    End Select
Else
    '文件不存在时，执行以下代码
    Set resultWK = Workbooks.Add                      '新添工作簿
    With resultWK
        .SaveAs FileLoc                               '保存新工作簿
        Do Until .Sheets.Count = 1                    '删除新工作簿中的表直到只有一个表
            .Sheets(1).Delete                         '删除表
        Loop
        DeleteID = .Sheets(1).Name                    '保存最后一个表的名称
    End With
End If
On Error Resume Next
rs.Close                                              '关闭记录集
On Error GoTo 0
rs.Open "select * from 工作表 order by 工作簿路径", cnn 临时数据簿     '按工作簿路径字段打开工作
                                                                      表记录集
If rs.RecordCount Then
    rs.MoveFirst                                      '记录集移动到第一条记录
    Do Until rs.EOF                                   '循环直到最后一条记录
        If strPreWB = "" Then                         '检测前一个打开工作簿的路径
            Set wk = Workbooks.Open(rs.Fields("工作簿路径"))      '打开工作簿
```

```
            strPreWB = wk.FullName                              '定义 strPreWB 字符串
        End If
        'strPreWB 与当前工作簿路径不一致时，需要打开该工作簿
        If strPreWB <> rs.Fields("工作簿路径") And strPreWB <> "" Then
            wk.Close
            Set wk = Workbooks.Open(rs.Fields("工作簿路径"))    '打开对应工作簿路径的工作簿
            strPreWB = wk.FullName                              '保存工作簿路径到 strPreWB
        End If
        '定义需要复制的工作表
        Set ws = wk.Sheets(Left(rs.Fields("工作表名"), Len(rs.Fields("工作表名")) -1))
        If ws.Name = DeleteID Then                              '检测工作表是否与 DeleteID 相同
            resultWK.Sheets(DeleteID).Name = DeleteID + "0"     '修改 DeleteID 表的名称
            DeleteID = DeleteID + "0"                           '修改 DeleteID
        End If
        ws.Copy before:=resultWK.Sheets(1)                      '复制工作表
        rs.MoveNext                                             '移动记录集指针到下一条
    Loop
    wk.Close                                                    '关闭工作簿
    resultWK.Sheets(DeleteID).Delete                            '删除 Delete 表
    Application.DisplayAlerts = True                            '显示警告消息
    resultWK.Save                                               '保存工作簿
    '根据语言设置，显示对应备份完成信息
    If languageset Then
        MsgBox "BackFile Process accomplished!", vbOKOnly + vbInformation
    Else
        MsgBox "备份文件完成!", vbOKOnly + vbInformation
    End If
Else
    '根据语言设置，显示没有任何表选中信息
    If languageset Then
        MsgBox "There is no sheet selected!", vbOKOnly + vbExclamation
    Else
        MsgBox "没有任何表被选中!", vbOKOnly + vbExclamation
    End If
End If
Exit_Sub:
Application.ScreenUpdating = True
Set ws = Nothing
Set wk = Nothing
Set rs = Nothing
End Sub
```

12.3.8　链接字符串与工作簿名获取过程代码设计

链接字符串与工作簿名获取过程是两个自定义函数。两个函数都接受一个字符串参数并返回一个字符串作为该函数的结果。这两个函数的功能及运行方式描述如下：

- 获取工作簿名：函数接受工作簿的完整路径，注意该参数是按值传递的。首先程序

将该参数字符串反向，再从反向字符串中获取“\”符号第一次出现的位置。然后程序从反向字符串左边第一个字符开始，取到“\”出现位置的所有字符。该获取的字符串即为工作簿名的反向字符串。将该获取得到的字符串反向后即为该函数的返回值。

- 获取链接字符串：函数接受一个 Excel 或 Access 文件的完整路径，然后函数返回链接到该文件的链接字符串。该函数可以接受 Excel 2007 或 Access 2007 文件作为数据库文件。

以下是两个函数的详细代码解释：

```
Public Function 获取工作簿名(ByVal strWKPath As String) As String
Dim strTemp As String, i As Integer
strTemp = StrReverse(strWKPath)           '反向工作簿文件路径
i = InStr(1, strTemp, "\")                '获取反向工作簿文件路径中第一个“\”符号出现的位置
strTemp = Left(strTemp, i -1)             '获取工作簿名的反向字符串
获取工作簿名 = StrReverse(strTemp)          '获取工作簿名
End Function

Public Function GetConnString(strFilePath As String) As String
'系统中只使用了两个文件作为数据库文件。一个是该加载宏文件，一个是 Access 2007 数据库文件
'所以这里的代码十分简单。只对文件后缀稍加判断就确认了相应的链接字符串格式
If LCase(Right(strFilePath, 5)) = "accdb" Then
    ‘文件为 Access2007 文件时的链接字符串
    GetConnString = "Provider=microsoft.ace.oledb.12.0;data source=" & strFilePath
Else
    '文件为本 Excel 2007 加载宏文件时的链接字符串
    GetConnString = "Provider=microsoft.ace.oledb.12.0;extended properties=Excel 12.0;data
source=" _
                & strFilePath
End If
End Function
```

12.4 拆分工作簿窗体设计

拆分工作簿窗体可以完成各项拆分设置，用户单击【开始拆分】按钮后，程序将利用这些设置完成拆分工作。设置工作包括拆分工作表位置设置、工作表分组组别设置、工作表分配设置，这些设置在窗体中都可以逐项被实现。

该窗体拆分时，只能对一个工作簿的工作表进行拆分。拆分出来的工作簿可以包含原工作簿的多个工作表。

12.4.1 窗体界面设计

窗体界面分为 3 大块。第一块是设置拆分工作簿的路径，第二块显示所有未分配工作表

列表，第三块用于设置分组组别以及工作表的组别分配设置。表 12-5 列出了该窗体中所有控件的控件名、控件类型及控件说明。如图 12-12 所示为该窗体的界面。

表 12-5　拆分工作簿窗口控件列表

控　件　名	控 件 类 型	控 件 说 明
Frame 拆分工作簿	框架	该控件用于包含设定需要拆分的工作簿的控件
txt 拆分工作簿	文本框	该文本框用于显示需要拆分的工作簿的位置
btn 浏览	按钮	单击该按钮后打开文件路径获取窗口。在窗口中选择了文件后，该文件的路径将被显示在拆分工作簿文本框中
Frame 未分配工作表	框架	该控件用于包含未分配工作表列表的控件
List 未分配表	ListView	该控件显示所有没有未分配的工作表
btn 添加	按钮	单击该按钮后，未分配表中被选中的表将被分配到右侧的列表中。所处的分类即为当前分组
Frame 已分配工作表	框架	该框架包含组别设置框，以及该组别中的工作表列表
comb 组别	复合框	该控件中显示了所有的工作表分组类别。单击【开始拆分】按钮后，拆分出来的新工作簿将使用该类别建立
btn 添加类别	按钮	该按钮用于将组别复合框中的新类别添加到数据库中，并刷新组别复合框
btn 删除	按钮	单击该按钮后，程序将把该分组下选中的工作表重新分配到未分配表中
btn 开始拆分	按钮	单击该按钮后，程序将开始执行拆分工作。在此之前，需要保证没有表未被分配

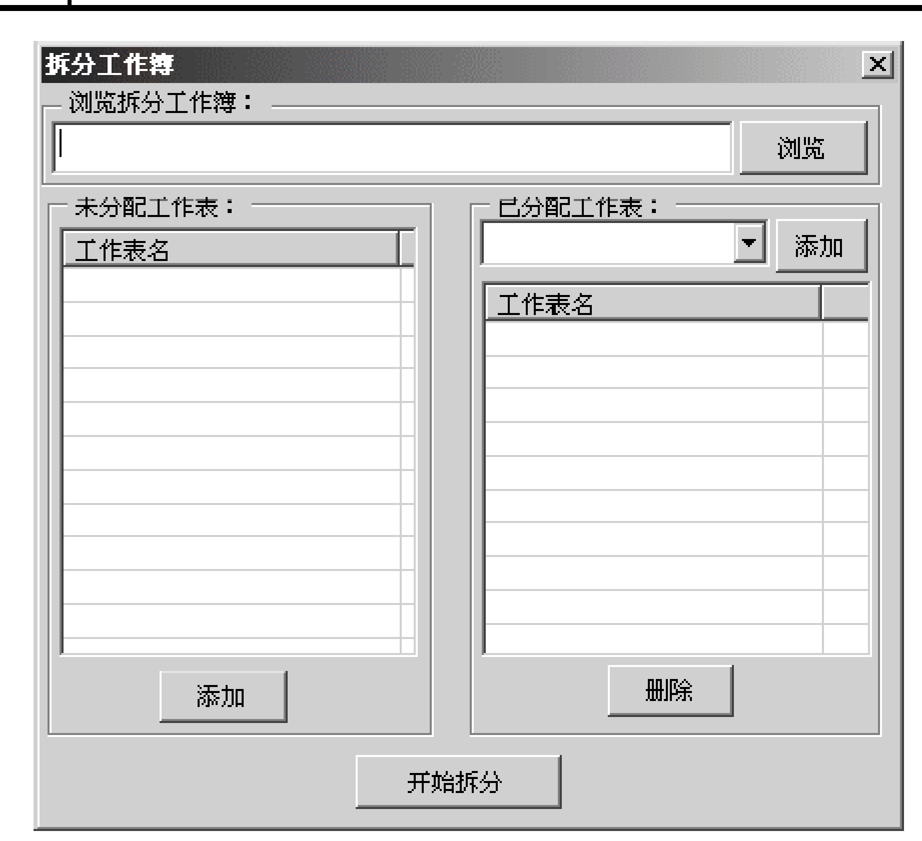


图 12-12　拆分工作簿窗体界面

制作该窗体的步骤如下：

（1）在 Excel 2007 的 VBE 开发环境中依次选择【插入】|【用户窗体】命令，在属性窗口中设置名称属性为“frm 拆分工作簿”，如图 12-13 所示。

（2）在工具箱中选择框架控件。在窗体中单击鼠标左键并拖动以产生适当大小的框架，随后复制该框架 2 份。在属性窗口中依次设置各个框架控件的 Caption 属性为“浏览拆分工作簿：”、“未分配工作表：”和“已分配工作表：”。名称属性依次设置为“Frame 拆分工作簿”、“Frame 未分配工作表”和“Frame 已分配工作表”。窗体的实际设计效果如图 12-14 所示。



图 12-13　拆分工作簿窗口属性设计



图 12-14　拆分工作簿窗体设计效果图

（3）在工具箱中选择文本框控件。在“浏览拆分工作簿”框架左侧插入一文本框。在属性窗口中修改名称属性为“txt 拆分工作簿”，SelectionMargin 属性为 False，如图 12-15 所示。

（4）在工具箱中选择按钮控件。在“浏览拆分工作簿”框架右侧插入一个按钮。在“未分配工作表”框架底部、“已分配工作表”框架右上侧及底部和窗体的底部各再插入一个按钮。然户在属性窗口中依次设置各个按钮的 Caption 属性为“浏览”、“添加”、“添加”、“删除”和“开始拆分”，并设置各个按钮的名称为“btn 浏览”、“btn 添加”、“btn 添加类别”、“btn 删除”和“btn 开始拆分”。

（5）在工具箱中选择复合框控件。在“已分配工作表”框架左上侧插入一复合框，随后在属性窗口中设置该复合框的名称属性为“comb 组别”，如图 12-16 所示。



图 12-15　设置文本框的 SelectionMargin 属性



图 12-16　复合框属性设计

（6）在工具箱中选择 ListView 控件。在“未分配工作表”框架和“已分配工作表”框架中各插入一个 ListView 控件，随后在属性窗口中依次设置两控件的名称属性为“List 未分配表”和“List 已分配表”。

12.4.2　变量定义与窗口激活事件代码设计

窗口中定义了一个变量 str。该变量存储需拆分工作簿文件的后缀，拆分后的工作簿将被保存为该后缀的工作簿文件。因为 Excel 2007 与之前版本的 Excel 文件的后缀不一样，根据该后缀名可以确定保存的文件类型和原文件类型一致。

窗口初始化时，需要清除数据库中存储的有关拆分工作簿的记录信息。在数据库中拆分表与拆分表组别两个表的记录都需要清除。程序通过 ADO 数据库链接的 Execute 方法执行删除命令操作，然后通过刷新 List 控件实现对 ListView 控件显示效果的重置。该过程的代码请见后续小节介绍。

```
Dim str 后缀  As String

Private Sub UserForm_Activate()
cnn 临时数据簿.Execute "delete * from  拆分表"
cnn 临时数据簿.Execute "delete * from  拆分表组别"
刷新 List 控件
End Sub
```

12.4.3　刷新 List 控件过程代码设计

刷新 List 控件过程用于设置窗口中两个 ListView 控件的显示效果以及标题行。该过程代码比较简单，以下是该过程的代码解释。

```
Private Sub  刷新 List 控件()
'设置 List 未分配表控件
With List 未分配表
    .Gridlines = True                                       '显示网格线
    .FullRowSelect = True                                   '允许整行选择
    .MultiSelect = True                                     '允许多行选择
    .LabelEdit = lvwManual                                  '单击项目时，不进入编辑状态
    .View = lvwReport                                       '设置显示模式
    .CheckBoxes = True                                      '是否显示项目选择框
    With .ColumnHeaders
        .Clear                                              '清除控件标题
        If languageset Then
            .Add Text:="SheetName", Width:=125              '英文标题显示
        Else
            .Add Text:="工作表名", Width:=125               '中文标题显示
        End If
    End With
End With
'设置 List 已分配表控件
With List 已分配表
    .Gridlines = True                                       '显示网格线
    .FullRowSelect = True                                   '允许整行选择
    .MultiSelect = True                                     '允许多行选择
    .LabelEdit = lvwManual                                  '单击项目时，不进入编辑状态
    .View = lvwReport                                       '设置显示模式
    .CheckBoxes = True
    With .ColumnHeaders
        .Clear                                              '清除控件标题
        If languageset Then
```

```
                .Add Text:="SheetName", Width:=125          '英文标题显示
            Else
                .Add Text:="工作表名", Width:=125           '中文标题显示
            End If
        End With
End With
End Sub
```

12.4.4　拆分工作簿文本框与浏览按钮代码设计

拆分工作簿文本框和浏览按钮共同完成设置拆分工作簿位置的工作。浏览按钮用于获取需拆分的工作簿的路径，并将该路径显示在拆分工作簿文本框中。当拆分工作簿文本框中数据发生变化时，将激发拆分文本框改变事件。

该事件过程首先清除了原来存储在数据库中的信息，以便存储新工作簿的信息资料。接着程序检测新指定文件路径下是否有该文件存在，当存在时，程序首先获取该工作簿所有工作表的名称。然后将工作表显示到 List 未分配表控件中，并将这些表名写入数据库中。新写入数据库的表名标记为“未分配”。图 12-17 是该过程的流程图。

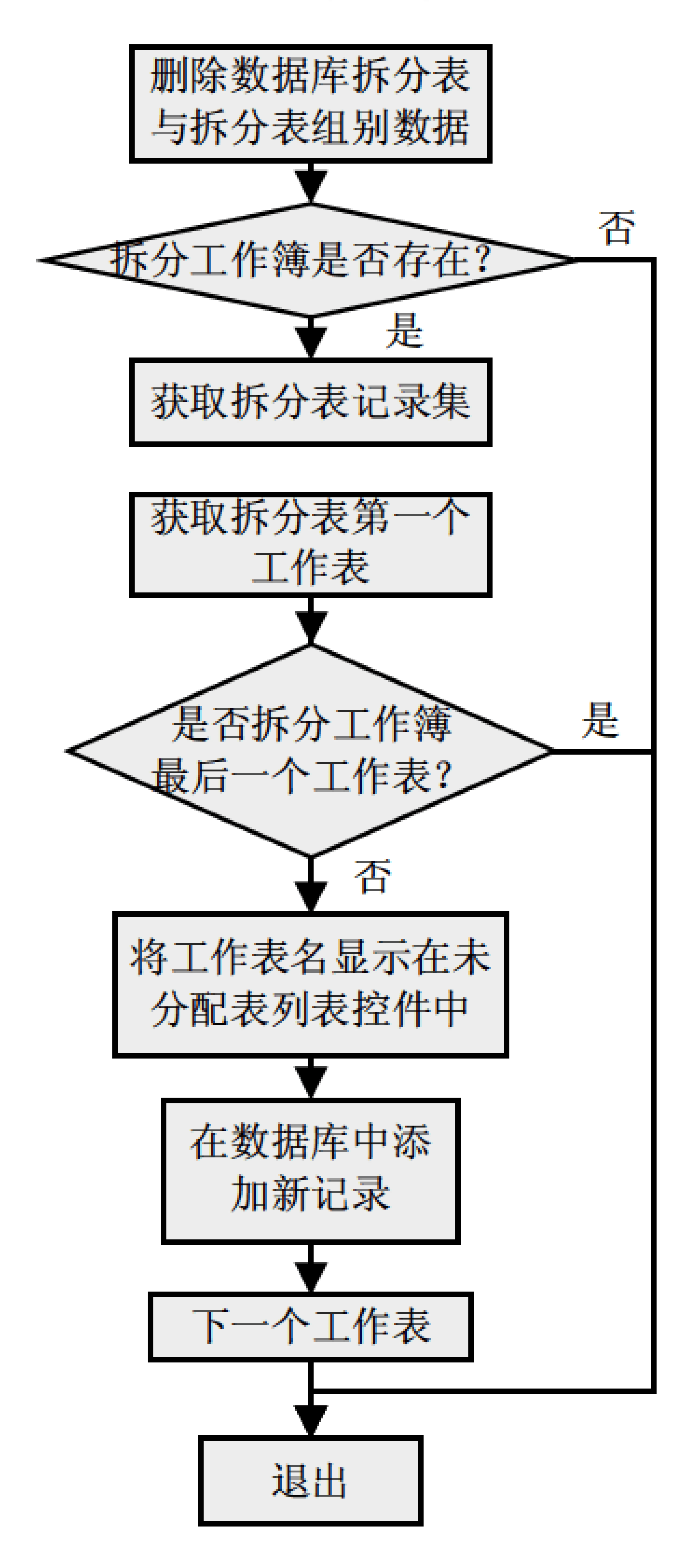


图 12-17　拆分工作簿文本框改变事件过程的流程图

以下是该事件的代码解释：

```
Private Sub txt 拆分工作簿_Change()
Dim strTemp As String
Dim catalog As New ADOX.catalog, table As New ADOX.table
cnn 临时数据簿.Execute "delete * from  拆分表"             '删除拆分表所有记录
cnn 临时数据簿.Execute "delete * from  拆分表组别"          '删除拆分表组别所有记录
strTemp = Dir(txt 拆分工作簿.Text)                          '查找拆分工作簿
On Error Resume Next
rs.Close
On Error GoTo 0
If strTemp <> "" Then                                                  '检测拆分工作簿是否存在
    catalog.ActiveConnection = GetConnString(txt 拆分工作簿.Text)        '获取拆分工作簿对应的
catalog 对象
    rs.Open "select * from  拆分表", cnn 临时数据簿, adOpenKeyset, adLockOptimistic   '打开记录集
    List 未分配表.ListItems.Clear                            '清除未分配表列表中所有项目
    For Each table In catalog.Tables                         '循环 Catalog 对象中所有表对象
        List 未分配表.ListItems.Add Text:=Left(table.Name, Len(table.Name) -1)        '添加新项目
        With rs
            rs.AddNew                                        '为记录集增加新记录
            rs.Fields("工作表名") = table.Name               '设置工作表名字段
            rs.Fields("组名") = "未分配"                      '设置组名字段
            rs.Update                                        '更新拆分表
        End With
    Next
End If
End Sub

Private Sub btn 浏览_Click()
'获取拆分工作簿的完全路径
str 拆分工作簿 = Application.GetOpenFilename("Excel2000-2007 file(*.xls;*.xlsx),*.xls;*.xlsx",
MultiSelect:=False)
txt 拆分工作簿.Text = str 拆分工作簿                          '将选择路径显示到拆分工作簿文本框中
End Sub
```

12.4.5　添加按钮单击事件代码设计

未分配工作表框架中的【添加】按钮在被单击时，将把未分配工作表列表中所有选中的工作表转移到右侧分组中。

过程首先检测右侧的组别是否有选择项。当没有选择项时，提示没选择分类后直接退出过程。当有分组时，程序逐个检测未分配工作表中的项目，当项目被选中时，在已分配表列表中添加该项目，然后将该项目在数据库中的组名标示为该组别，最后程序把未分配工作表列表中选中项目移除后退出过程。该过程的流程图如图 12-18 所示。

限于篇幅，流程图中关于删除未分配表列表中选中项目过程没有列出具体的操作流程。下面将介绍该过程流程。该过程是一个 Do…Loop 循环，其循环变量是项目的索引号。

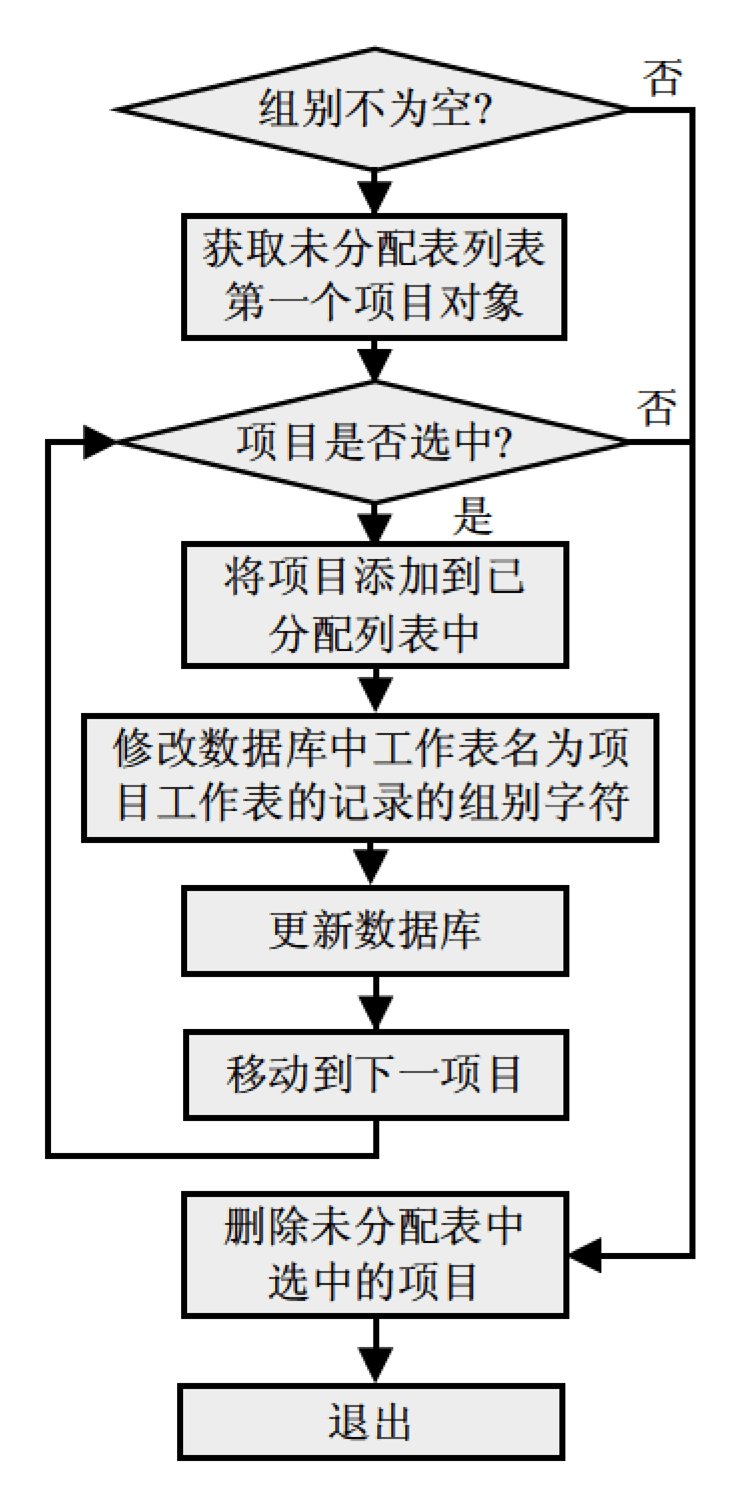


图 12-18 【添加】按钮单击事件流程图

循环开始时，程序从分配表列表第一个项目开始。如果该项目未被选中，则跳到下一个项目，循环变量将自动加 1，以将项目指向下一条。如果项目被选中，程序将删除该项目，但是循环变量不用加 1。因为再次回到循环开始时，未分配表中项目数量已经发生了改变，而已删除项目的下一条项目（即当前检测项目）的索引号等于删除项目的索引号。因而这种情况下检测的项目索引号不需要修改。当最后索引号大于列表项目数时说明已经检测完毕，即可退出循环。

以下是该过程的代码解释：

```
Private Sub btn 添加_Click()
Dim itemlist As ListItem, strSQL As String
If comb 组别.Text = "" Then                                  '检测组别复合框是否为空
    MsgBox "没有选择分类！"                                   '提示没有选择组别分类
    Exit Sub                                                  '退出过程
End If
For Each itemlist In List 未分配表.ListItems                  '循环未分配表列表中所有项目
    If itemlist.Checked Then                                  '检测项目是否被选中
        List 已分配表.ListItems.Add Text:=itemlist.Text       '将选中项目添加到已分配表列表中
        '设定数据库查询字符串，查询祖名为未分配，工作表名为项目值的记录
        strSQL = "select * from 拆分表 where 组名='未分配' and 工作表名='" & itemlist.Text & "$'"
        On Error Resume Next
        rs.Close
        On Error GoTo 0
        With rs
```

```
                .Open strSQL, cnn 临时数据簿, adOpenKeyset, adLockOptimistic    '打开记录集
                .MoveFirst                                    '移动到记录集首条记录
                .Fields("工作表名") = itemlist.Text & "$"      '设定工作表名字段
                .Fields("组名") = comb 组别.Text                '设定组名字段
                .Update                                       '更新记录集
            End With
            rs.Close                                          '关闭记录集
        End If
Next
i = 1                                                         '初始化检测项目索引号
Do Until i > List 未分配表.ListItems.Count                      '循环直到索引号超过项目总数
    If List 未分配表.ListItems(i).Checked Then                  '检测未分配表第 i 项是否被选中
        List 未分配表.ListItems.Remove I                        '将第 i 项删除
    Else
        i = i + 1                                             '计算下一个检测项目的索引号
    End If
Loop
End Sub
```

12.4.6　组别复合框改变事件代码设计

组合复合框允许用户选择不同的组别分类。当组别分类发生改变时，需要将新组别下所有已分配工作表显示在列表中。复合框改变事件完成的工作即为该任务。

程序首先从数据库拆分表中获取组名为新选定组别的工作表名字段记录集，接着清除已分配表列表的所有项目，以便于重新添加新项目下被分配的表。然后程序循环记录集中所有记录，将工作表名字段作为已分配表列表新项目添加进去。该过程的流程图如图 12-19 所示。

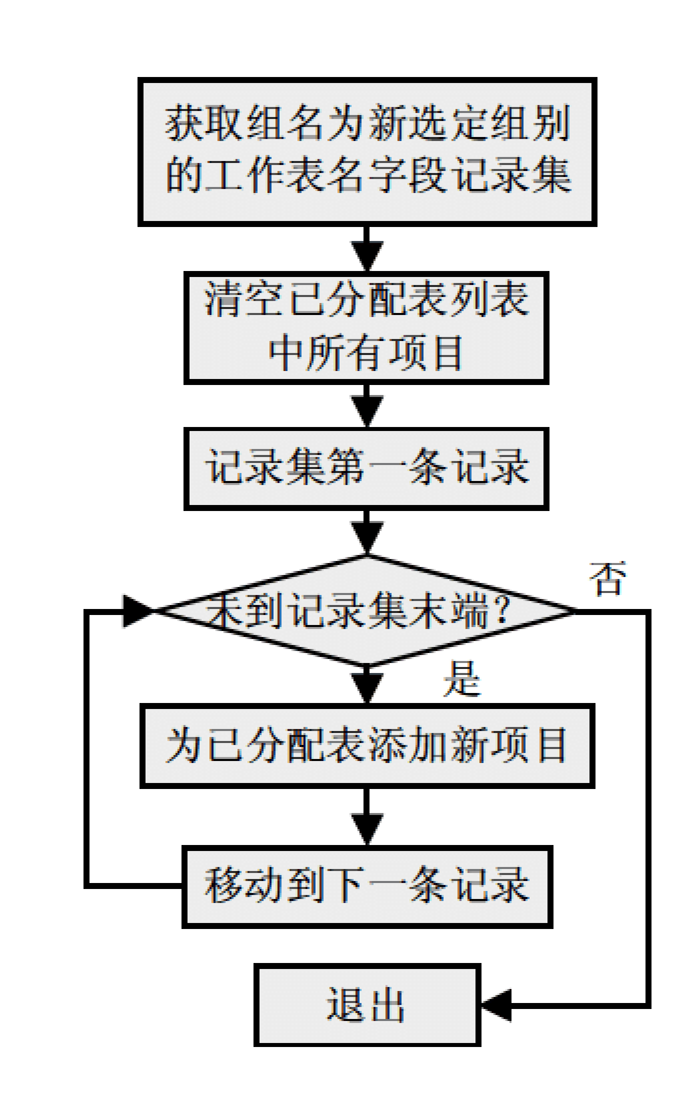


图 12-19　组别复合框改变事件流程图

该过程的代码解释如下：

```
Private Sub comb 组别_Change()
Dim strSQL As String, strTemp As String
On Error Resume Next
rs.Close
'获取组名为新选定组别的工作表名字段记录集
strSQL = "select 工作表名 from 拆分表 where 组名='" & comb 组别.Text & "'"    '设定查询字符串
rs.Open strSQL, cnn 临时数据簿, adOpenKeyset, adLockOptimistic                '获取记录集
List 已分配表.ListItems.Clear                                                '清除列表项目
If rs.RecordCount Then
    Do Until rs.EOF                                        '到达记录集末端时，终止循环
        strTemp = rs.Fields("工作表名")                     '获取工作表名字段
        strTemp = Left(strTemp, Len(strTemp) -1)           '去掉工作表名最后的$符号
```

```
            List 已分配表.ListItems.Add Text:=strTemp                    '为已分配表添加新项目
            rs.MoveNext                                                  '移动到下一条记录
        Loop
End If
On Error GoTo 0
End Sub
```

12.4.7 添加按钮单击事件

【添加】按钮用于将在组别复合框新输入的组别添加到数据库中。程序首先检测该组名在数据库中是否已经存在。当已经存在时，直接退出过程即可。当检测完所有记录完仍然没有找到该组别时，程序将该组别添加进数据库中。图 12-20 所示的是该过程的流程图。

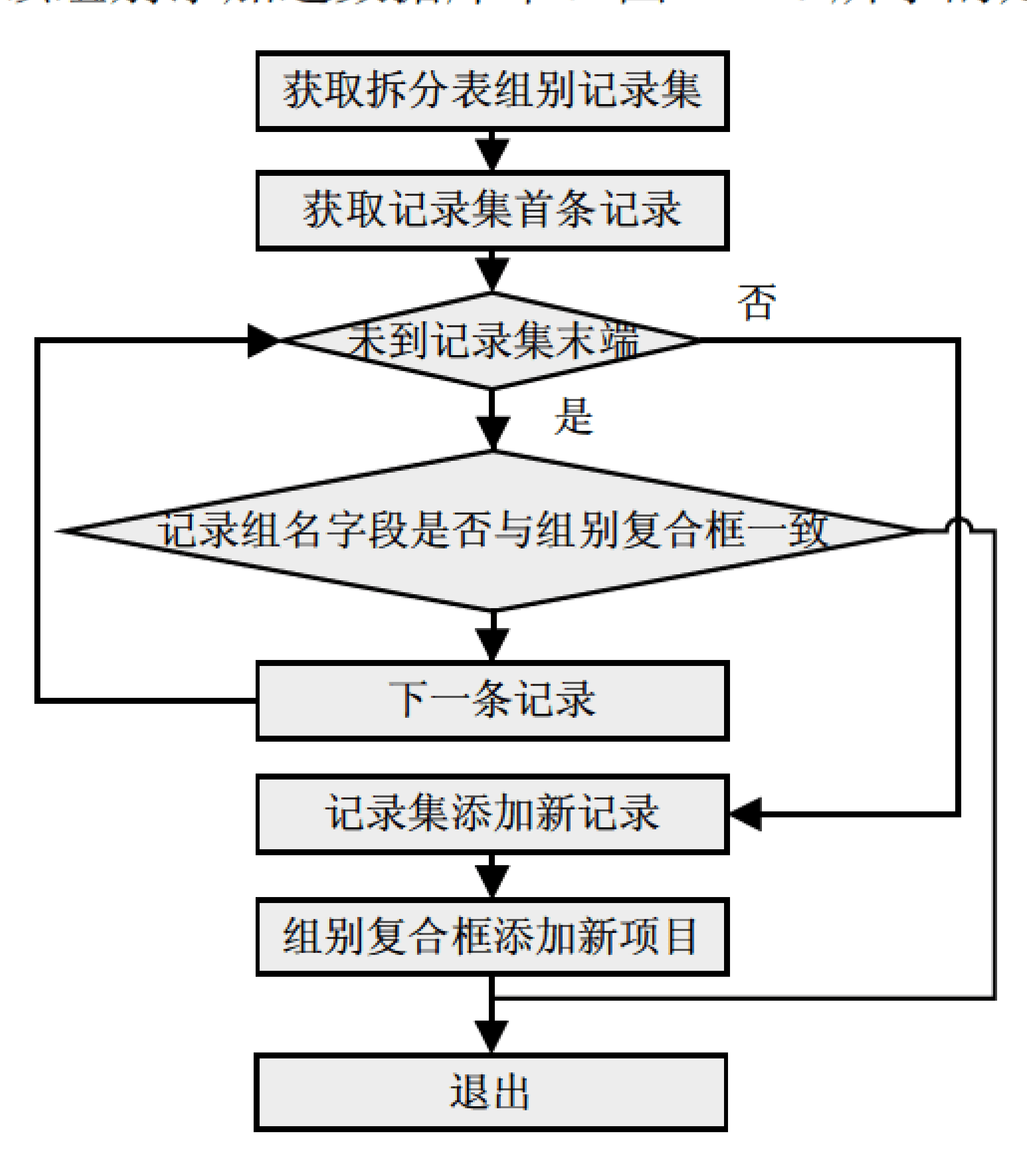


图 12-20　添加类别过程流程图

以下是该过程的代码解释：

```
Private Sub btn 添加类别_Click()
On Error Resume Next
rs.Close                                                     '关闭记录集
On Error GoTo 0
'打开到拆分表组别的记录集
rs.Open "select * from  拆分表组别", cnn 临时数据簿, adOpenKeyset, adLockOptimistic
If rs.RecordCount Then
    Do Until rs.EOF                                          '循环到记录集最末端
        If comb 组别.Text = rs.Fields("组名") Then           '检测组别是否在记录中已经存在
            MsgBox "该组别已经存在！ ", vbInformation + vbOKOnly, "组别已经存在"    '提示存在
            Exit Sub
        End If
        rs.MoveNext                                          '移动到下一条记录
    Loop
```

```
End If
With rs
    .AddNew                                       '添加新记录
    .Fields("组名") = comb 组别.Text               '设置新记录的组别字段
    .Update                                       '更新拆分表组别数据
End With
comb 组别.AddItem comb 组别.Text                   '在组别复合框中添加该新组别
comb 组别.Value = comb 组别.Text                   '设置组别复合框的显示值
MsgBox "分配类别添加成功！"                         '提示添加成功
rs.Close                                          '关闭记录集
End Sub
```

12.4.8　删除按钮单击事件代码设计

删除操作是未分配工作表框架中【添加】按钮的一个反向操作。该按钮将把已分配工作表列表中所有选中项目移动到未分配工作表列表中，并且修改数据库中这些项目对应工作表的组别为“未分配”。

程序首先逐个检测已分配表列表中的项目。当项目被选中时，程序将把该项目添加到未分配表列表中，然后通过数据库链接执行一个更新查询完成数据库中组别字段的更新操作，最后程序将根据更新后的数据库标示，重新显示已分配列表。

以下是该单击事件过程的代码解释：

```
Private Sub btn 删除_Click()
Dim itemlist As ListItem, strSQL As String
On Error Resume Next
rs.Close                                                  '关闭记录集
On Error GoTo 0
For Each itemlist In List 已分配表.ListItems                '逐个循环已分配表的项目
    If itemlist.Checked Then
        '设置更新查询字符串
        strSQL = "update  拆分表  set  组名='未分配' where  工作表名='" & itemlist.Text & "$'"
        List 未分配表.ListItems.Add Text:=itemlist.Text     '在未分配表列表中添加该项目
        cnn 临时数据簿.Execute strSQL                       '执行更新查询
    End If
Next
With List 已分配表.ListItems
    .Clear                                                '清空已分配表项目
     '获取组别为当前选择组别的拆分表记录集
    rs.Open "select * from  拆分表  where  组名='" & comb 组别.Text & "'", cnn 临时数据簿
    If rs.RecordCount Then
        Do Until rs.EOF                                   '循环到记录集末端后终止
            .Add Text:=Left(rs.Fields("工作表名"), Len(rs.Fields("工作表名")) -1)   '为列表添加新项目
            rs.MoveNext                                   '移动到下一条记录
        Loop
    End If
End With
End Sub
```

12.4.9 开始拆分按钮单击事件代码设计

【开始拆分】按钮的单击事件过程是系统拆分模块的核心过程。单击该按钮后，程序将对已经保存到数据库中的信息完成拆分工作。而这些信息是用户在窗口中设置的一个数据备份。该过程的流程较为复杂，以下分别使用文字与流程图加以说明。

在对程序流程加以说明之前，需要将该过程的设计思路讲述一下。首先打开需拆分的工作簿，然后按照用户设置的拆分方式即各个工作表分别保存到哪几个新工作簿中，依次建立几个新工作簿，并把对应工作表复制到各个工作簿中即完成了拆分工作。以上说到的拆分设置都被保存到了数据库中，因而需要读取数据库信息。

以下是该过程的流程文字说明：

程序首先从数据库的拆分表中获取组名为“未分配”的所有记录。如果该记录集中仍然有记录，说明用户还没有对所有工作表做相应设置，此时需要提示用户完成分类操作，并直接退出过程，否则程序继续后续代码。随后程序开启了需要拆分的工作簿、从拆分表组别表中获取记录集以及获取工作簿的后缀名。这几个变量和对象在拆分时将被调用。

接着程序检测拆分表组别记录集是否有记录。值得注意的是，这里的组别对应的就是新建的工作簿的名称。当记录集存在记录时，程序将循环所有记录。每一次循环中，程序都会从数据库中获取当前组别下所包含的所有工作表名，并将这些工作表从拆分工作簿中复制到新建的工作簿中。为了确保新建的工作簿只包含指定名称的工作表，循环体中还包含了如何剔除新建工作簿时的多余表代码。如图 12-21 所示的是该过程的流程图。

以下是该过程的详细代码解释：

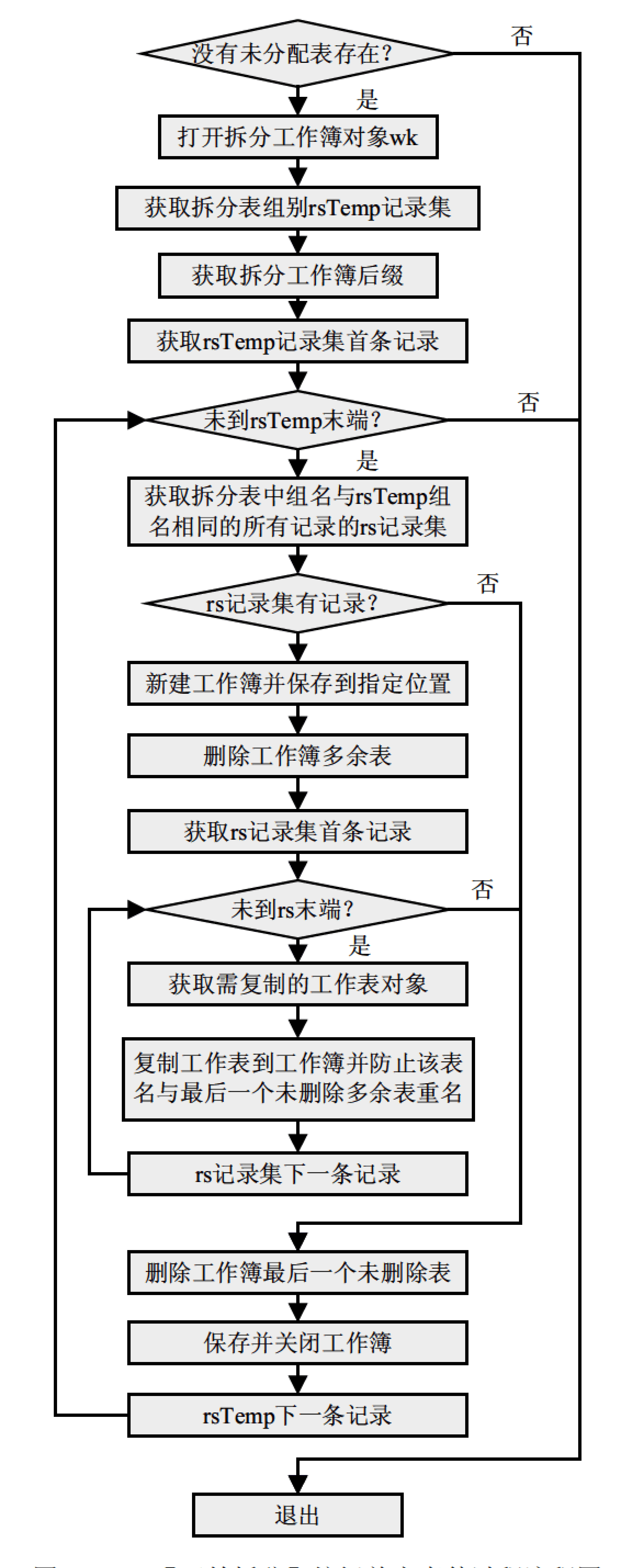


图 12-21 【开始拆分】按钮单击事件过程流程图

```
Private Sub btn 开始拆分_Click()
Dim wk As Workbook, ws As Worksheet, wk2 As Workbook
Dim rsTemp As New ADODB.Recordset, strSQL As String, DeleteID As String
Application.ScreenUpdating = False
Application.DisplayAlerts = False
On Error Resume Next
rs.Close
On Error GoTo 0
rs.Open "select * from 拆分表 where 组名='未分配'"          '设置查询字符串
If rs.RecordCount Then                                   '检测记录集是否为空
    MsgBox "还有未被分配的表存在！请将这些未分配表设置到一个新的分类中！", vbInformation +
vbOKOnly, "存在未分配表"
    Exit Sub
End If
Set wk = Workbooks.Open(txt 拆分工作簿)                    '获取工作簿对象
rsTemp.Open "select * from 拆分表组别", cnn 临时数据簿      '获取拆分表组别记录集
str 后缀 = 文件后缀(txt 拆分工作簿)                          '获取工作簿后缀名称
If rsTemp.RecordCount Then                               '检测拆分表组别记录集是否为空
    Do Until rsTemp.EOF                                  '循环所有拆分表组别记录集
        '设置查询记录集字符串，该记录集中所有记录的组名应该等于 reTemp 记录集的组名
        strSQL = "select * from 拆分表 where 组名='" & rsTemp.Fields("组名") & "'"
        On Error Resume Next
        rs.Close
        On Error GoTo 0
        rs.Open strSQL, cnn 临时数据簿, adOpenKeyset, adLockOptimistic   '获取条件查询记录集
        If rs.RecordCount Then                                          '检测记录集是否为空
            Set wk2 = Workbooks.Add                                     '新增工作簿
            wk2.SaveAs 保存文件名(txt 拆分工作簿, rsTemp.Fields("组名"))  '按组名保存新工作簿
            Do Until wk2.Sheets.Count = 1                '检测工作簿是否只有一个表
                wk2.Sheets(1).Delete                     '删除工作簿第一个工作表
            Loop
            DeleteID = wk2.Sheets(1).Name                '记录没有删除的工作表的名称
            Do Until rs.EOF                              '循环 rs 记录集所有记录
                '获取需要复制工作表的工作表对象
                Set ws = wk.Worksheets(Left(rs.Fields("工作表名"), Len(rs.Fields("工作表名")) -1))
                If ws.Name = DeleteID Then               '检测该工作表是否与需删除表名称重名
                    wk2.Sheets(DeleteID).Name = DeleteID + "0"     '修改需删除表的名称
                    DeleteID = DeleteID + "0"                      '记录需删除表的新名称
                End If
                rs.MoveNext                              '将记录指针移到下一条
                ws.Copy before:=wk2.Sheets(1)            '复制工作表到新工作簿中
            Loop
         End If
         wk2.Sheets(DeleteID).Delete                     '删除原来不能删除的多余工作表
         wk2.Save                                        '保存新工作簿
         wk2.Close                                       '关闭新工作簿
         rsTemp.MoveNext                                 '将 rsTemp 记录指针移到下一条
    Loop
End If
```

```
wk.Close
Application.ScreenUpdating = True
Application.DisplayAlerts = True
MsgBox "保存成功！"
End Sub
```

12.4.10 文件后缀与保存文件名过程代码设计

设计中会用到文件后缀与保存文件名两个自定义函数。文件后缀函数根据传入的工作簿文件路径获取该文件的后缀名。系统中使用该函数获取拆分工作簿的后缀，以确保拆分出来后保存所得的工作簿与原工作簿文件类型一致。保存文件名函数根据传入的拆分工作簿文件路径和组别名称（即保存文件的名称），确定保存文件位置。拆分出来的工作簿文件将位于拆分工作簿相同文件夹中。以下是这两个函数的代码解释：

```
Private Function 文件后缀(ByVal fileName As String) As String
Dim i As Integer
fileName = StrReverse(fileName)                    '反向文件路径字符串
i = InStr(1, fileName, ".")                        '获取反向路径字符串第一个“.”位置
fileName = StrReverse(Left(fileName, i -1))        '获取后缀名字符串
文件后缀 = filename                                 '设置函数返回值
End Function

Private Function 保存文件名(filePath As String, fileName As String) As String
Dim i As Integer
filePath = StrReverse(filePath)              '反向拆分工作簿路径字符串
i = InStr(1, filePath, "\")                  '获取反向拆分工作簿路径字符串第一个“.”位置
filePath = Left(StrReverse(filePath), Len(filePath) -i + 1)     '获取排除文件名后的路径字符串
保存文件名 = filePath & fileName & "." & str 后缀                  '设置保存工作簿的完成路径
End Function
```

12.5 选择工作簿窗体设计

选择工作簿窗口用于获取需要合并工作表的所有工作簿。在该窗口中，用户也可以打开其他不需要合并工作表的工作簿。用户只需要在进入选择工作表之前，将这些工作簿取消选中即可。用户选择的工作簿以及所有的选中状态信息都会被记录到数据库中，以便在窗口重显时读取。

12.5.1 窗口界面设计

窗口中共包含了 2 个框架控件、2 个复选框控件、2 个单选按钮控件、1 个 ListView 控件和 2 个按钮控件，如图 12-22 所示。窗口的控件大致可以划分为 4 块，即工作簿显示区域、选中设置区域、选择语言区域和按钮区。这些控件的具体说明如表 12-6 所示。

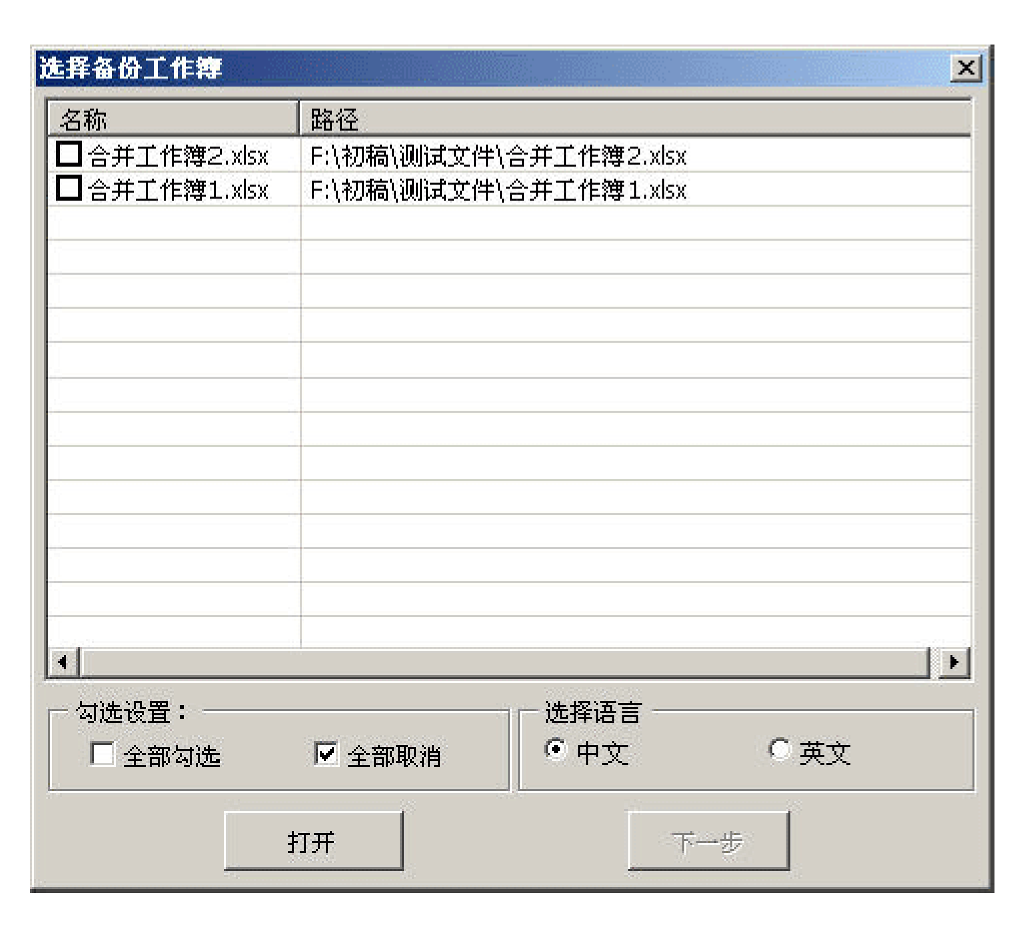


图 12-22　选择备份工作簿界面设计

表 12-6　选择备份工作簿窗口控件列表

控　件　名	控 件 类 型	控 件 说 明
List 工作簿	ListView	该控件用于显示用户已打开工作簿。在该控件中用户还可以勾选需要备份工作表的工作簿，以进入下一步中设置备份的工作表
Frame 选中设置	框架	该控件包含了两个设置工作簿选中操作的复选框控件
chk 全部选中	复选框	选择该复选框时，将自动选中所有 ListView 控件中所有工作簿。当用户手动将 ListView 控件中的工作簿都选中时，该复选框将自动被选中
chk 全部取消	复选框	选择该复选框时，将自动取消所有 ListView 控件中已勾选工作簿。当用户手动将 ListView 控件中的选中工作簿都取消时，该复选框将自动被选中
frame 选择语言	框架	该框架用于设置语言显示。它包含了两个单选按钮
op 中文	单选按钮	选中该单选按钮时，将把当前语言设置为中文。该窗口的显示将立即更新为中文
op 英文	单选按钮	选中该单选按钮时，将把当前语言设置为英文。该窗口的显示将立即更新为英文
btn 打开	按钮	单击该按钮后，将弹出一个文件路径获取窗口。在该窗口中用户选择一个或多个工作簿后，这些工作簿的名称和路径将会被记录在数据库中。并在窗口列表中显示出来
btn 下一步	按钮	单击该按钮后，将进入工作表选择窗口中。用户可以在该窗口中设置各个工作簿中需要备份的工作表

建立该窗口的步骤如下：

（1）在 Excel 2007 的 VBE 开发环境中依次选择【插入】|【用户窗体】命令。在属性面板中修改窗口名称为“frm 选择工作簿”，如图 12-23 所示。

（2）在工具箱中选择 ListView 控件。在窗体的上部插入一个 ListVeiw 控件后，在属性窗口中将其名称属性修改为“List 工作簿”，CheckBoxes 属性设置为 True，如图 12-24 所示。该属性设置为 True 时，ListView 控件的每个项目前都会出现一个复选框。




图 12-23　选择工作簿窗体属性设计



图 12-24　ListView 控件属性设计示意图

（3）在工具箱中选择框架控件。在靠近 ListView 控件的下方连续插入两个框架控件。在属性窗口中将控件的名称属性依次设置为“Frame 勾选设置”和“frame 选择语言”，然后将其 Caption 属性依次设置为“勾选设置：”和“选择语言：”，如图 12-25 所示。

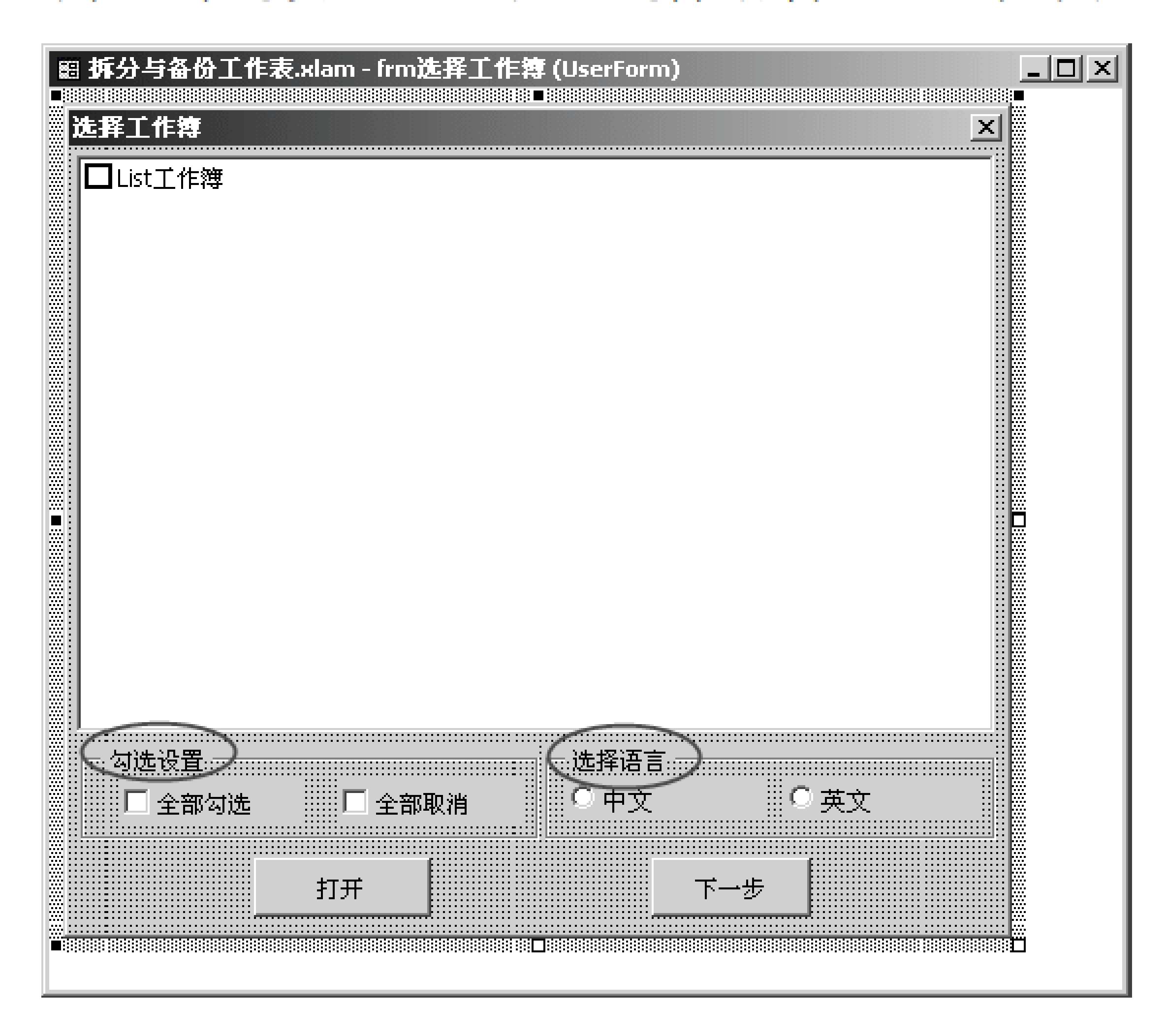


图 12-25　选择工作簿窗体设计效果

（4）在工具箱中选中复选框控件，在“勾选设置”框架中连续插入两个复选框。随后在属性窗口中设置名称属性依次为“chk 全部勾选”和“chk 全部取消”，Caption 属性依次设置为“全部勾选”和“chk 全部取消”。

（5）在工具箱中选择单选框控件，在“选择语言”框架中连续插入两个复合框。随后在属性窗口中设置其名称属性依次为“op 中文”和“op 英文”，Caption 属性依次设置为“中文”和“英文”。

（6）在工具箱中选择按钮控件并在窗口底部连续插入两个按钮。随后在属性窗口设置名称属性依次为“btn 打开”和“btn 下一步”，Caption 属性依次设置为“打开”和“下一步”。

12.5.2　窗口事件代码设计

窗口相关的过程包括了 3 个事件代码，分别是窗口初始化事件、窗口激活事件和窗口卸载事件。这 3 个事件的功能描述如下：

- 窗口初始化事件：该事件首先建立到本工作簿的链接，该链接在设置语言显示时被使用，然后程序初始化了一些公共变量的初始值与控件的初始状态。
- 窗口激活事件：当窗口被激活时，程序将调用刷新窗口语言显示过程刷新窗口显示，然后再刷新 ListView 控件，其中分标题刷新和列表项目刷新，最后设置了窗口中各个控件状态。
- 窗口卸载事件：窗口卸载时，程序通过一个 If 语句确保了备份工作表中的 3 个窗口只有一个被显示。

以下是这 3 个事件过程的代码解释：

```
Private Sub UserForm_Initialize()
On Error Resume Next
cnn 主工作簿.Open GetConnString(ThisWorkbook.FullName)  '获取到当前工作簿的链接
op 中文.Value = True                                     '设置中文设置单选框的状态
languageset = False                                      '将语言设置为中文显示(中为为 False)
btn 下一步.Enabled = False                                '设置下一步按钮的可用状态
On Error GoTo 0
End Sub

Private Sub UserForm_Activate()
刷新窗体语言显示 Me                                  '刷新窗口语言显示
ListView 标题刷新                                     '刷新 ListView 控件的标题
刷新工作簿列表                                        '刷新 ListView 控件的项目
设置控件状态                                          '设置窗口中其他控件的状态
End Sub

Private Sub UserForm_Terminate()
If UserForms.Count = 3 Then                          '窗口中有 3 个窗口时，卸载保存位置窗口
    Unload frm 保存位置
ElseIf UserForms.Count = 2 Then
    Unload frm 选择工作表                              '窗口中有两个窗口时，卸载选择工作表窗口
End If
Unload Me                                            '卸载选择工作簿窗口
End Sub
```

12.5.3　工作簿列表控件代码设计

相关工作簿列表控件的代码包含了 3 个过程，分别是：列表项目勾选事件、列表项目选择事件、ListView 标题刷新过程。这 3 个过程的作用描述如下：

- 列表项目勾选事件：当在工作簿列表中选中了某个项目时，事件被激发。该事件将只完成一个任务，即重新设置控件状态。程序并没有在这里重新设置数据库中对应该被选中工作簿的是否选中字段，而是选择在单击【下一步】按钮时集中修改。
- 列表项目选择事件：选择列表项目时，程序只设置了该项目的提示文本。该文本将在鼠标在该项目上停留一定时间后被显示。
- ListView 标题刷新过程：标题刷新过程重新设置了 ListView 控件的显示参数，并且

重置了该控件的标题。

以下是这 3 个过程的代码解释：

```
Private Sub List 工作簿_ItemCheck(ByVal Item As MSComctlLib.ListItem)
设置控件状态                                         '调用设置控件状态过程
End Sub

Private Sub List 工作簿_ItemClick(ByVal Item As MSComctlLib.ListItem)
Item.TooltipText = Item.SubItems(1)                  '设置项目提示文本
End Sub

Private Sub ListView 标题刷新()
With Me.List 工作簿
    .Gridlines = True                                '显示网格线
    .FullRowSelect = True                            '允许整行选择
    .MultiSelect = True                              '允许多行选择
    .LabelEdit = lvwManual                           '单击项目时，不进入编辑状态
    .View = lvwReport                                '设置控件显示模式
    With .ColumnHeaders
        .Clear                                       '清除控件标题
        If languageset Then
            .Add Text:="Name"                        '设置英文显示的名称栏标题
            .Add Text:="Path", Width:=255            '设置英文显示的路径栏标题
        Else
            .Add Text:="名称"                        '设置中文显示的名称栏标题
            .Add Text:="路径", Width:=255            '设置中文显示的路径栏标题
        End If
    End With
End With
End Sub
```

12.5.4 选中设置与语言设置框架代码设计

选中设置和语言设置框架中包含了 2 个复选框和 2 个单选按钮。这些按钮被选中时都需要执行一部分代码完成特定任务。2 个复选框被选中时，程序需要修改 ListView 控件中项目的选中状态以及下一部按钮的可用状态。2 个单选按钮被选中时，需要重新显示窗口语言以及 ListView 标题的语言。这些代码都比较简单。这里不再列出各个过程的流程图，以下将通过文字加以介绍。

- ❑ 全部勾选和全部取消复选框单击事件：程序首先检测全部勾选（全部取消）复选框的值。如果为真，说明全部勾选（全部取消）复选框被选中。此时，程序将首先修改全部取消（全部勾选）复选框的值，然后程序循环工作簿列表中所有项目，对于未选中（选中）的项目将设置为选中（未选中）。最后程序将设置【下一步】按钮可用（不可用）。
- ❑ 中文和英文单选按钮单击事件：中文（英文）单选按钮被单击时，首先设置语言设

置公共变量为假（真），然后程序刷新了控件中所有控件的 Caption 属性，最后程序将 ListView 控件的显示标题也修改为中文（英文）。

以下是这 4 个单击事件过程的代码解释：

```
Private Sub chk 全部勾选_Click()
Dim itemlist As ListItem
If chk 全部勾选.Value Then                        '检测全部复选框是否选中
    chk 全部取消.Value = False                    '取消选中全部复选框
    For Each itemlist In List 工作簿.ListItems
        If itemlist.Checked = False Then          '检测项目是否未选中
            itemlist.Checked = True               '设置项目被选中
        End If
    Next
    btn 下一步.Enabled = True                     '设置下一步按钮可用
End If
Set itemlist = Nothing
End Sub

Private Sub chk 全部取消_Click()
Dim itemlist As ListItem
If chk 全部取消.Value Then                        '检测全部复选框是否取消选中
    chk 全部勾选.Value = False                    '选中全部复选框
    For Each itemlist In List 工作簿.ListItems
        If itemlist.Checked Then                  '检测项目是否选中
            itemlist.Checked = False              '设置项目未选中
        End If
    Next
    btn 下一步.Enabled = False                    '设置下一步按钮不可用
End If
Set itemlist = Nothing
End Sub

Private Sub op 英文_Click()
languageset = True                                '设置语言设置公共变量为真
刷新窗体语言显示 Me                                '刷新窗口所有控件 Caption 属性为英文显示
ListView 标题刷新                                  '刷新 ListView 控件标题栏文本为英文显示
End Sub

Private Sub op 中文_Click()
languageset = False                               '设置语言设置公共变量为假
刷新窗体语言显示 Me                                '刷新窗口所有控件 Caption 属性为中文显示
ListView 标题刷新                                  '刷新 ListView 控件标题栏文本为中文显示
End Sub
```

12.5.5　打开与下一步按钮代码设计

窗口中共包含了两个按钮：【打开】按钮和【下一步】按钮。这些按钮被单击后，分别

完成各自的功能。这些按钮的功能描述如下：

- 打开按钮单击事件：调用 Excel 2007 的内部函数 GetOpenFileName。该函数将开启一个【打开文件】对话框，用户选择了文件后，函数返回选择文件的路径。程序中将这个返回值保存到了一个 Variant 数据类型的变量中。因为用户可能选择了多个文件，此时这个返回值将是一个数组。然后程序将把这些工作簿保存到数据库中，最后刷新工作簿列表中的项目。
- 下一步按钮单击事件：单击【下一步】按钮后，程序首先保存选中工作簿到数据库中，然后隐藏了选择工作簿窗口，最后将选择工作表窗口显示出来。

以下是这两个按钮单击事件过程的代码解释：

```
Private Sub btn 打开_Click()
On Error GoTo ExitSub_Handle
'打开工作簿路径获取窗口，并将选择工作簿保存到公共变量中
arr 选择工作簿 = Application.GetOpenFilename("Excel2000-2007 file(*.xls;*.xlsx),*.xls;*.xlsx",
MultiSelect:=True)
Application.ScreenUpdating = False                    '禁止 Excel 程序自动刷新显示
保存选择工作簿                                        '保存打开工作簿到数据库中
刷新工作簿列表                                        '刷新工作簿列表项目显示
Application.ScreenUpdating = True                     '恢复 Excel 程序自动刷新
ExitSub_Handle:
On Error GoTo 0
End Sub

Private Sub btn 下一步_Click()
保存选中工作簿                                        '调用保存选中工作簿过程
Me.Hide                                               '隐藏选择工作簿窗口
frm 选择工作表.Show                                   '显示选择工作表窗口
End Sub
```

12.5.6　设置控件状态过程代码设计

设置控件状态过程用于完成对窗口中控件状态的设置。窗口中设置控件状态的控件数量不多。包括【下一步】按钮的可用状态、【全部勾选】与【全部取消】复选框的勾选状态。

程序首先计算出工作簿列表中被勾选项目的数量。然后当该数大于 0 时，【下一步】按钮被设置为可用。当该数等于列表的项目总数时，【全部勾选】复选框被设置为勾选，而【全部取消】复选框被设置为未勾选。当该数等于 0 时，【全部取消】复选框被设置为勾选，而【全部勾选】复选框被设置为未勾选。此时【下一步】按钮也被设置为不可用。如果该数处于 0 与工作簿列表项目总数之间时，两个复选框都被设置为未选中。

以下是该过程的代码解释：

```
Private Sub 设置控件状态()
Dim itemlist As ListItem, checkCount As Integer
For Each itemlist In List 工作簿.ListItems            '循环所有工作簿列表中的项目
    If itemlist.Checked Then                          '检测项目是否选中
```

```
            checkCount = checkCount + 1                 '累计选中项目数
        End If
Next
If checkCount Then btn 下一步.Enabled = True            '选中项目数大于 0 时，下一步按钮可用
If checkCount = List 工作簿.ListItems.Count Then         '检测项目是否被全部选中
        chk 全部勾选.Value = True                        '选中全部复选框
        chk 全部取消.Value = False                       '取消选中全部复选框
End If
If checkCount = 0 Then                                   '检测项目是否全部取消选中
        chk 全部勾选.Value = False                       '选中全部复选框
        chk 全部取消.Value = True                        '取消选中全部复选框
        btn 下一步.Enabled = False                       '下一步按钮不可用
End If
'检测被勾选项目数是否落在 0 与总项目数间
If checkCount > 0 And checkCount < List 工作簿.ListItems.Count Then
        chk 全部勾选.Value = False                       '选中全部复选框
        chk 全部取消.Value = False                       '取消选中全部复选框
End If
End Sub
```

12.6　选择工作表窗体设计

选择工作表窗口用于获取需要备份的各个工作表。用户需要在该窗口中选择对应的工作簿，然后在该工作簿中选择相应工作表。这些需要备份保存的工作表将会被数据库记录，这些数据在进入下一步设置工作簿备份位置过程中被调用。

12.6.1　窗口界面设计

窗口包含了 1 个框架控件、2 个复选框控件、1 个复合框控件、1 个 ListView 控件和 2 个按钮。表 12-7 对窗口中包含的所有控件进行了具体的说明。如图 12-26 所示为该窗体的界面。

表 12-7　选择工作表窗体控件列表

控　件　名	控 件 类 型	控 件 说 明
List 选定工作簿	复合框	该复合框中包含了前面所有被选中的工作簿。复合框中选择工作簿后，未分配工作表列表将自动刷新
List 工作表	ListView	该控件用于显示当前选定工作簿下未分配的所有工作表
Frame 勾选设置	框架	该框架包含两设置选中方式复选框
chk 全部勾选	复选框	用户单击该复选框后，程序自动选中工作表列表中所有工作表
chk 全部取消	复选框	用户单击该复选框后，程序自动取消工作表列表中所有工作表的选中
btn 工作表上一步	按钮	用户单击该按钮后，可以退回到上一步工作簿设置对话框中
btn 工作表下一步	按钮	用户单击该按钮后，将打开合并工作簿位置设置对话框



图 12-26　选择工作表窗体界面

建立该窗口的步骤如下：

（1）在 Excel 2007 的 VBE 开发环境中依次选择【插入】|【用户窗体】命令，然后在属性窗口中设置该窗口的名称为“frm 选择工作表”，如图 12-27 所示。

（2）在工具箱中选择标签控件并在窗体中连续插入两个标签控件。在属性窗口依次设置两个标签的 Caption 属性为“当前工作簿：”和“选择工作表：”，如图 12-28 所示。



图 12-27　选择工作表窗体属性设计



图 12-28　选择工作表窗体设计效果

（3）在工具箱中选择复合框控件并在窗体当前工作簿标签下插入一个复合框控件。在属性窗口中设置该控件的名称为“List 选定工作簿”，SelectionMargin 属性设置为 False，如图 12-29 所示。



图 12-29　设置复合框控件的 SelectionMargin 属性

（4）在工具箱中选择 ListView 控件并在“选定工作簿”复合框下方插入一个 ListView 控件。在属性窗口中设置该控件的名称为“List 工作表”，如图 12-30 所示。

（5）在工具箱中选择框架控件。在刚插入的 ListView 控件右侧插入一个框架控件。随后在属性窗口中设置该框架的名称为“Frame 勾选设置”，如图 12-31 所示。



图 12-30　ListView 控件属性设计



图 12-31　框架控件属性设计

（6）在工具箱中选中复选框控件，在刚插入的框架控件中连续插入两个复选框控件。随后在属性窗口中设置两控件的名称属性依次为“chk 全部勾选”和“chk 全部取消”。

（7）在工具箱中选择按钮控件，在刚插入的框架控件下面连续插入两个按钮控件。在属性窗口中依次设置两按钮的 Caption 属性为“上一步”和“下一步”，名称属性依次设置为“btn 工作表上一步”和“btn 工作表下一步”。

12.6.2　窗口激活与卸载事件代码设计

窗口代码中包含了两个有关窗口事件的过程。它们是窗口激活事件和窗口卸载事件。这两过程的功能描述如下：

- 窗口激活事件：当窗口被重新激活时，需要刷新窗体语言显示及标题、重新刷新选定工作簿复合框的项目。程序首先调用刷新窗体语言显示公共过程完成窗体语言显示刷新工作簿，再调用刷新标题刷新了本窗体中 ListView 控件的标题，然后打开到数据库中工作簿表的记录集，该记录集的是否选定字段为 True。接着将该记录集中所有工作簿路径依次作为列表的项目添加进去。
- 窗口卸载事件：当窗口被卸载时，程序需要保证备份模块中包含的 3 个窗口都被关闭。程序首先检测窗口数量，数量为 3 时，说明保存工作簿窗口被打开了，首先需要关闭该窗口，然后依次卸载该窗口自身和选择工作簿窗口。这里没有必要检测窗口数为 2 或 1 时的情况。

以下是这两个事件的代码解释：

```
Private Sub UserForm_Activate()
刷新窗体语言显示 Me                                    '刷新窗体语言显示
刷新标题                                               '刷新 ListView 控件的标题
'获取到工作簿表的记录集，该记录集的是否选定字段为 True
rs.Open "select * from [工作簿] where 是否选定=TRUE", cnn 临时数据簿, adOpenKeyset,
adLockOptimistic
With List 选定工作簿
    .Clear                                             '清除列表所有项目
    Do Until rs.EOF                                    '当到达记录集末端时，结束循环
        .AddItem rs.Fields("工作簿路径")               '为列表添加新项目
        rs.MoveNext                                    '将记录移动到下一条
    Loop
    .ListIndex = 0                                     '默认显示值为第一条项目
End With
```

```
Set rs = Nothing
End Sub

Private Sub UserForm_Terminate()
If UserForms.Count = 3 Then                           '检测是否有 3 个窗体被同时显示
    Unload frm 保存位置                                '卸载保存位置窗口
End If
Unload Me                                             '卸载本窗口
Unload frm 选择工作簿                                   '卸载选择工作簿窗口
End Sub
```

12.6.3 复合框改变事件代码设计

工作簿复合框中包含了所有在选择工作簿窗口中选中的工作簿。当用户在该控件中改变选择后，程序需要刷新工作表列表项目显示。该刷新显示工作一方面要将该工作簿的所有工作表的名称体现出来，另一方面还需要设置各个工作表的选定状态。如图 12-32 所示是该过程的流程图。

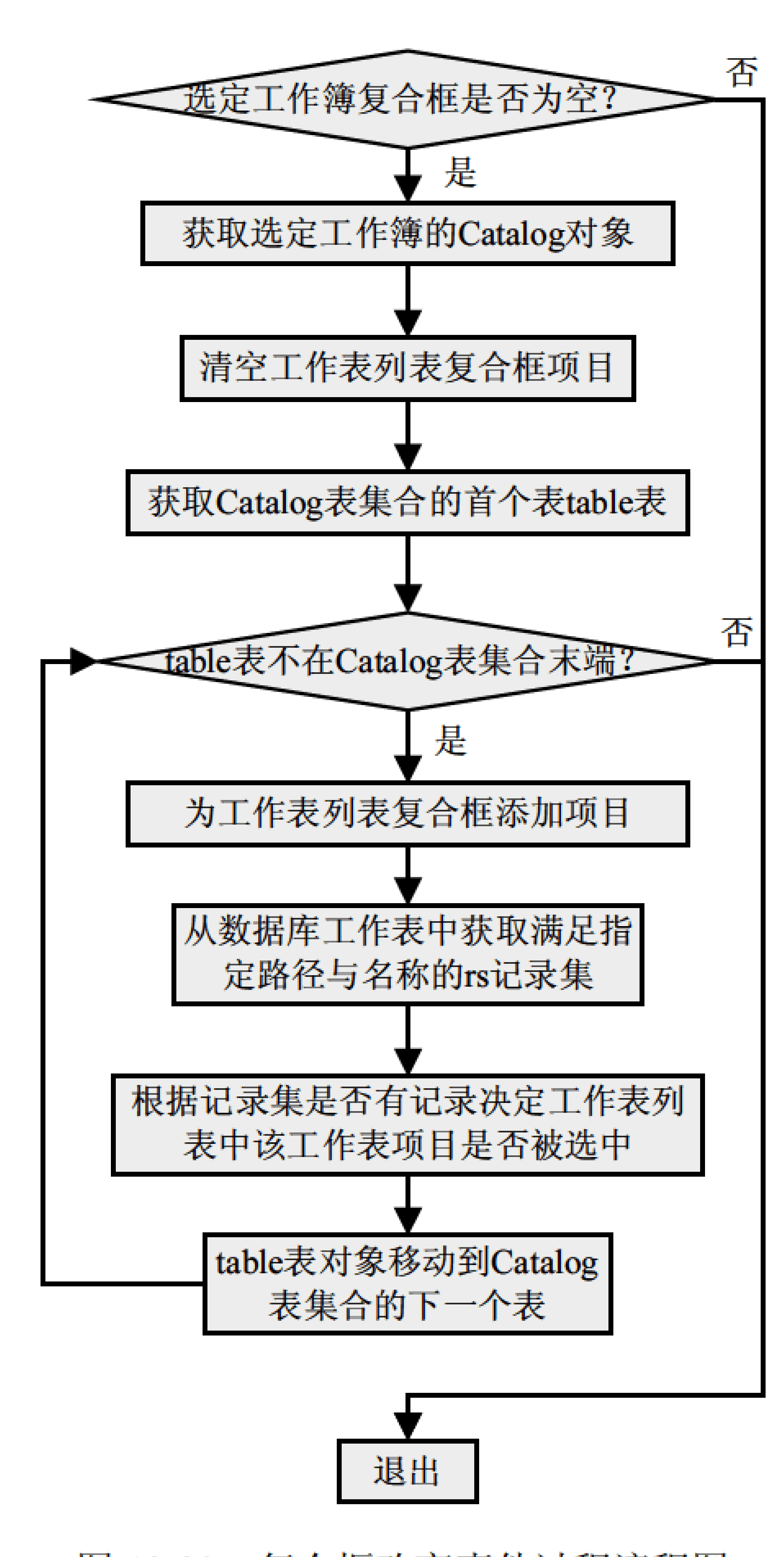


图 12-32　复合框改变事件过程流程图

以下是该过程的代码解释：

```
Private Sub List 选定工作簿_Change()
If Trim(List 选定工作簿.Text) = "" Then Exit Sub          '选定工作簿复合框为空时，退出过程
On Error Resume Next
Dim catalog As New ADOX.catalog, table As New ADOX.table
Dim strSQL As String
'获取选定工作簿的 Catalog 对象并且清空工作表列表中所有项目
catalog.ActiveConnection = GetConnString(List 选定工作簿.Text)    '获取 Catalog 对象
List 工作表.ListItems.Clear                                     '清空工作表列表项目
For Each table In catalog.Tables                                '循环选定工作簿的所有表
    With List 工作表.ListItems.Add(Text:=Left(table.Name, Len(table.Name) -1))
                                                                '为工作表列表添加项目
        strSQL = "select * from [工作表] where  工作簿路径='" & List 选定工作簿.Text & _
                "' and  工作表名='" & table.Name & "'"          '生成条件查询字符串
        rs.Close                                                '关闭记录集
        rs.Open strSQL, cnn 临时数据簿, adOpenKeyset, adLockOptimistic      '打开记录集
        If rs.RecordCount > 0 Then                              '检测记录集是否有记录
            .Checked = True                                     '设置新项目为被选中状态
        Else
            .Checked = False                                    '设置新项目为未选中
        End If
    End With
Next
设置表名显示状态                                                '设置窗口其他控件的显示状态
On Error GoTo 0
Set table = Nothing
Set catolog = Nothing
End Sub
```

12.6.4　工作表列表、选中设置与按钮代码设计

该小节包含了窗口工作表列表、勾选设置框架中控件与两按钮的代码。这些代码都比较简单且代码不多，因而将其归纳到一个小节加以介绍。下面将分别介绍这几个控件代码的功能：

- 工作表列表项目勾选事件：用户在选中了工作表列表时，将意味着该工作表将会被最终合并到备份工作簿中。此时需要将该工作表的名称以及所属工作簿的路径保存到数据库中，以便在备份工作簿工作中调用。最后程序还需要重新设置窗口中其他控件的显示状态。
- 全部勾选、全部取消复选框单击事件：勾选设置框架中包含的这两个复选框被单击时，程序需要按该设置完成项目勾选设置。其工作的流程类似，这里只对全部选中的工作流程加以说明，首先程序检测【全部勾选】复选框的值是否为真，为真时，程序将【全部取消】复选框值设置为假。然后循环检测工作表列表中所有项目，项目未被选中时，修改该项目为选中并通过调用添加删除选定项过程保存该设置。
- 上一步、下一步按钮单击事件：单击【上一步】按钮将退回到工作簿选择窗口中，重新选择需备份工作表的工作簿。单击【下一步】按钮将进入备份工作簿位置设置

对话框中。

以下是这些过程的代码解释：

```
Private Sub List 工作表_ItemCheck(ByVal Item As MSComctlLib.ListItem)
添加删除选定项  Item                                    '将选定项目信息写入数据库中
设置表名显示状态                                         '设置窗口其他控件的显示状态
End Sub

Private Sub chk 全部勾选_Click()
Dim itemlist As ListItem
If chk 全部勾选  Then                                    '检测全部勾选复选框值是否为真
    chk 全部取消.Value = False                           '设置全部取消复选框值为假
    For Each itemlist In List 工作表.ListItems           '循环工作表列表中所有项目
        If itemlist.Checked = False Then                 '检测项目 Checked 属性是否为假
            itemlist.Checked = True                      '设置项目为被选中
            添加删除选定项  itemlist                     '将该项目的信息写入数据库
        End If
    Next
End If
End Sub

Private Sub chk 全部取消_Click()
Dim itemlist As ListItem
If chk 全部取消  Then                                    '检测全部取消复选框值是否为真
    chk 全部勾选.Value = False                           '设置全部勾选复选框值为假
    For Each itemlist In List 工作表.ListItems           '循环工作表列表中所有项目
        If itemlist.Checked Then                         '检测项目 Checked 属性是否为真
            itemlist.Checked = False                     '设置项目为未选中
            添加删除选定项  itemlist                     '将该项目的信息从数据库中删除
        End If
    Next
End If
End Sub

Private Sub btn 工作表上一步_Click()
frm 选择工作簿.Show                                      '显示选定工作簿窗口
Me.Hide                                                  '隐藏本窗口
End Sub

Private Sub btn 工作表下一步_Click()
Unload Me                                                '卸载本窗口
frm 保存位置.Show                                        '显示保存位置窗口
End Sub
```

12.6.5 刷新标题过程代码设计

刷新标题过程用于重新设置选择工作表窗口中 ListView 控件的显示设置与标题显示。该刷新标题过程和先前各个包含 ListView 控件中的刷新标题过程类似。以下是该过程的代码解释：

```
Private Sub 刷新标题()
With List 工作表
    .Gridlines = True                    '显示网格线
    .View = lvwReport                    '设置显示模式
    .FullRowSelect = True                '允许整多行选择
    .MultiSelect = True                  '允许多行选择
    .LabelEdit = lvwManual               '单击项目时不进入编辑状态
End Sub
```

12.6.6　设置表名显示状态过程代码设计

设置表名显示状态过程用于设置窗口选中设置中两个复选框的选中状态，该过程仅仅只是从工作表列表中项目的选中数量来决定选中设置框架中复选框的值。过程首先通过一个 For 循环计算出工作表列表中被选中项目的数量，然后通过几个 If 判断语句确定两个复选框的值。如图 12-33 所示的是该过程的流程图。

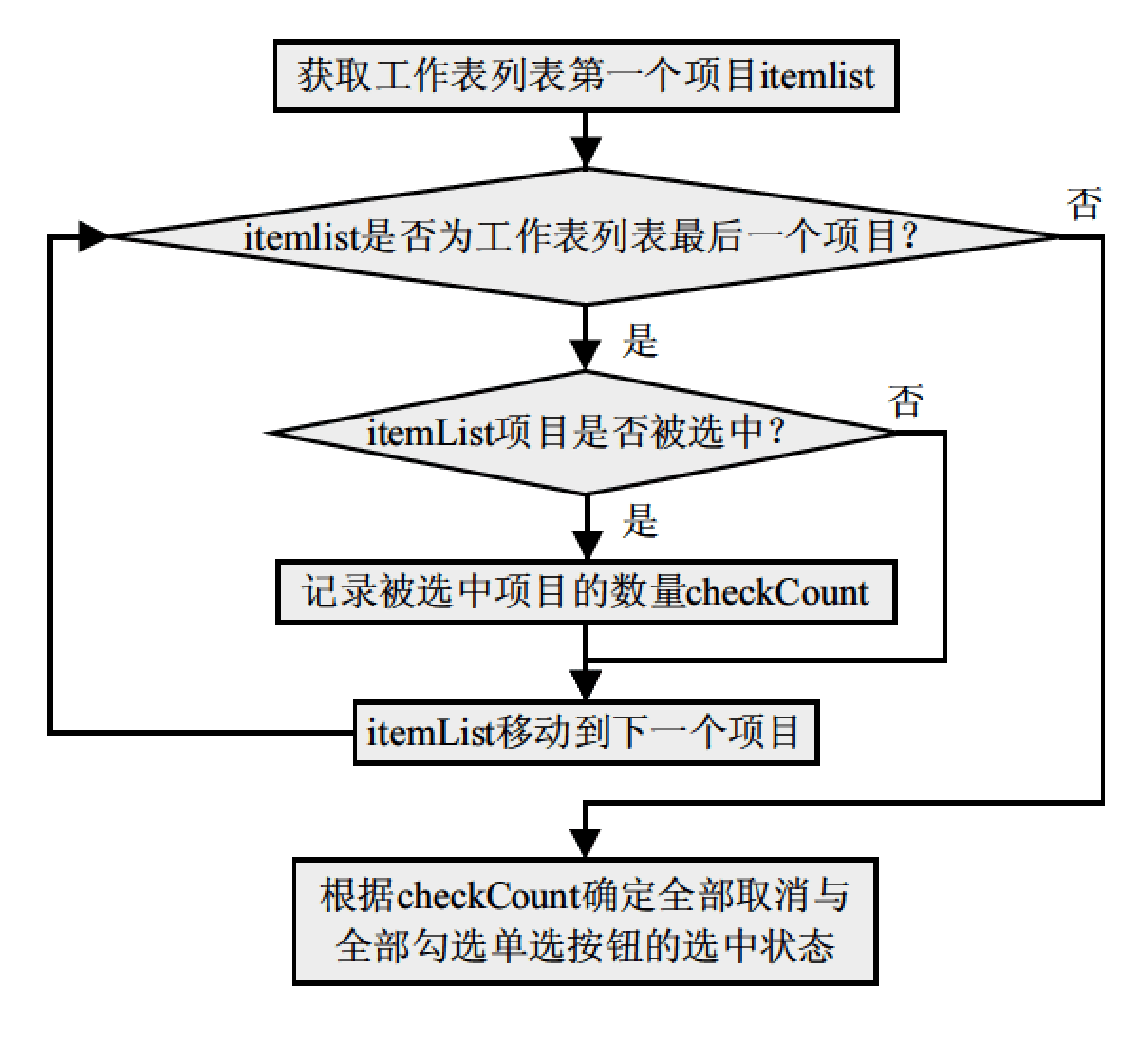


图 12-33　设置表名显示状态过程流程图

以下是该过程的代码解释：

```
Private Sub 设置表名显示状态()
Dim itemlist As ListItem, checkCount As Integer
For Each itemlist In List 工作表.ListItems            '循环工作表列表中所有项目
    If itemlist.Checked Then                           '检测项目是否被选中
        checkCount = checkCount + 1                    '记录被选中项目
    End If
Next
If checkCount = List 工作表.ListItems.Count Then       '判断是否全部项目被选中
    chk 全部勾选.Value = True                          '选中全部复选框
    chk 全部取消.Value = False                         '取消选中全部复选框
End If
If checkCount = 0 Then                                 '检测是否全部项目未选中
```

```
      chk 全部勾选.Value = False                         '选中全部复选框
      chk 全部取消.Value = True                          '选中全部复选框
End If
'判断被勾选项目数是否在 0 和项目总数量间
If checkCount > 0 And checkCount < List 工作表.ListItems.Count Then
      chk 全部勾选.Value = False                         '取消选中全部复选框
      chk 全部取消.Value = False                         '选中全部复选框
End If
End Sub
```

12.6.7 添加删除选定项过程代码设计

添加删除选定项过程根据传递列表项目对象被选中情况，对数据库中该项目的信息进行相应处理。当项目被选中时，将在数据库中添加该项目。当取消选中时，过程将该记录从数据库中删除。

程序首先从数据库的工作表中获取一记录集，该记录集的工作簿路径字段为当前选定工作表所处的工作簿，而工作表名字段即为该工作表名称。然后程序检测当前项目的选定状态，当被选定时，将向该记录集添加新记录，否则将该记录删除。该过程的流程图如图 12-34 所示。

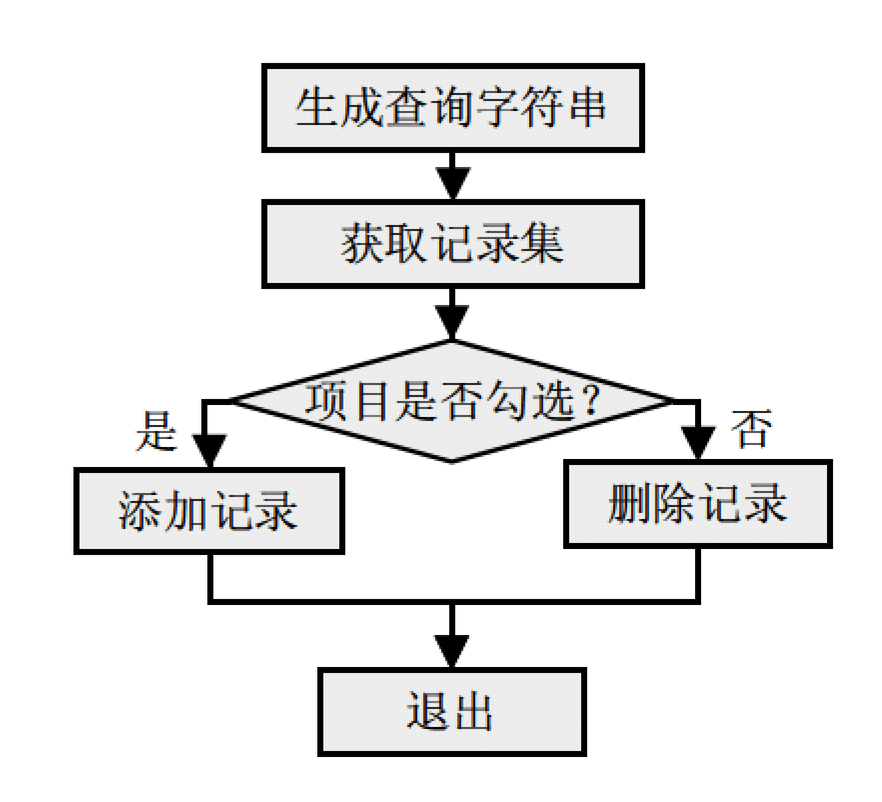


图 12-34　添加删除选定项目过程流程

```
Private Sub 添加删除选定项(itemlist As ListItem)
Dim strSQL As String
On Error Resume Next
strSQL = "from [工作表] where 工作簿路径='" & List 选定工作簿.Text & _
          "' and 工作表名='" & itemlist.Text & "$'"                    '生成查询字符串
rs.Close                                                              '关闭记录集
rs.Open "select * " & strSQL, cnn 临时数据簿, adOpenKeyset, adLockOptimistic   '打开记录集
If itemlist.Checked = True Then                                       '检测项目是否被
选定
      On Error GoTo 0
      rs.AddNew                                                       '添加新记录
      rs.Fields("工作簿路径") = List 选定工作簿.Text                     '设置工作簿路径
字段的值
      rs.Fields("工作表名") = itemlist.Text & "$"                      '设置工作表名字
段的值
      rs.Update                                                       '更新记录表
Else
      rs.Delete                                                       '删除记录
      rs.Update                                                       '更新记录表
End If
Set rs = Nothing
End Sub
```

12.7　保存文件窗口设计

保存文件窗口用于设置备份工作簿的保存位置。在该窗口中包含了一列表，该列表显示了所有用户已经选择需要备份的工作表。用户根据该列表可以再次检查需要备份的工作表。

在窗口中有一个备用文件名复合框，该复合框列出了所有源表工作簿的非重复名称。当用户命名备份工作簿时需要以源工作簿名称为基础时，可以从中选择。单击【浏览】按钮设置备份位置时，程序设置了一个默认保存文件名。该文件名使用了备份文件名加上当前时间，从而保证文件名的非重复。

12.7.1　窗口界面设计

窗口中包含了 2 个框架控件、3 个标签控件、1 个 ListView 控件、1 个文本框控件、1 个复合框控件和 3 个按钮控件。表 12-8 给出了这些控件的具体说明。如图 12-35 所示为该窗口的界面。

表 12-8　保存文件窗口控件列表

控　件　名	控 件 类 型	控 件 说 明
Frame 已选工作表	框架	该控件用于包含所有显示需备份工作表列表信息的控件。框架中只有一个 ListView 控件
LabelSelectedSheet	标签	该控件在已选工作表框架中用于显示提示信息
ListViewResult	ListView	该控件用于显示在选择工作表步骤中用户选择的所有工作表的信息
Frame 备份文件设置	框架	该控件包含了所有用于设置备份工作簿保存位置的控件
LabelBackUpLoc	标签	显示设置保存位置的提示信息
txtFileLoc	文本框	该控件显示用户设置的备份工作簿保存位置
btnBrowse	按钮	该控件打开一个文件路径获取窗口
LabelDefFileName	标签	该标签显示备用文件名的提示信息
ListFileName	复合框	该复合框包含所有的备用文件名的名称。这些名称是从用户选定工作簿名称中获取的
btnPreStepLoc	按钮	单击该按钮将回到上一步设置工作表窗口中
btnOKLoc	按钮	单击该按钮后，程序将按照用户的设置完成保存备份工作簿任务



图 12-35　保存文件窗口界面

制作该窗口的步骤如下：

（1）在 Excel 2007 的 VBE 开发环境下依次选择【插入】|【用户窗体】命令，在属性窗口中设置新建立窗口的名称属性为“frm 保存位置”，如图 12-36 所示。

（2）在工具箱中选择框架控件。在窗口中依次插入两个框架控件。随后在属性窗口依次设置这两个框架控件的名称为“Frame 已选工作表”和“Frame 备份文件设置”，Caption 属性依次设置为“已选工作表查看：”和“备份文件设置：”，如图 12-37 所示。



图 12-36　保存位置窗体属性设计

图 12-37　保存位置窗体设计效果

（3）在工具箱中选择标签控件。在“已选工作表：”框架的上部插入一个标签控件，然后在“备份文件设置：”框架中连续插入两个标签控件。随后在属性窗口中依次设置这 3 个标签控件的名称属性为 LabelSelectedSheet、LabelBackUpLoc 和 LabelDefFileName，Caption 属性依次设置为“已选定工作表(如需修改请回上一步)：”、“选择备份存储位置：”和“备选文件名：”。

（4）在工具箱中选择 ListView 控件。在“已选定工作表：”框架中插入一个 ListView 控件。随后在属性窗口中设置该控件的名称属性为 ListViewResult，如图 12-38 所示。

（5）在工具箱中选择文本框控件。在“备份文件设置：”框架中插入一个文本框控件。随后在属性窗口中设置该控件的名称属性为 txtFileLoc，SelectionMargin 属性设置为 False，如图 12-39 所示。

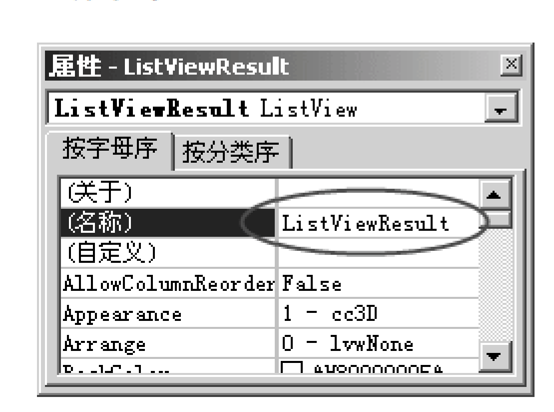


图 12-38　ListView 控件属性设计

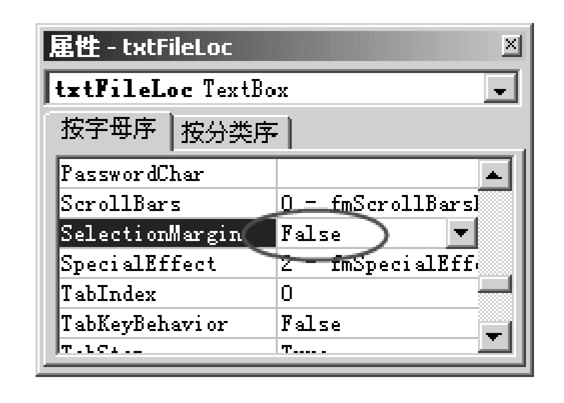


图 12-39　设置文本框控件的 SelectionMargin 属性

（6）在工具箱中选择复合框控件。在“备份文件设置：”框架的“备选文件名：”标签右侧插入一个复合框控件。随后在属性窗口中设置该复合框的名称属性为 ListFileName，SelectionMargin 属性设置为 False。

（7）在工具箱中选择按钮控件。在“备份文件设置”框架的文本框右侧插入一个按钮控件，然后在窗口底部再插入两个按钮控件。随后在属性窗口中依次设置这 3 个按钮的名称属性为 btnBrowse、btnPreStepLoc 和 btnOKLoc。

12.7.2 窗口事件与 ListView 事件代码设计

本小节将窗口事件和 ListView 控件的事件代码放置在一起加以介绍。ListView 控件事件代码很少，因而不再单独加以介绍。本部分包含了窗口激活事件、窗口卸载事件以及 ListView 控件项目单击事件。在窗口中还使用到了一个局部变量 m_fileLoc，该变量用于保存用户设置的保存文件名位置信息。以下是几个事件的具体功能介绍：

- 窗口激活事件：窗口激活时，需要根据用户设置的语言设置刷新窗口语言显示和更新 ListView 控件的标题显示，然后程序从数据库中读取已选定需要备份的工作表的信息到 ListView 控件，最后程序将默认的备选文件名添加到备选文件名复合框中。
- 窗口卸载事件：当用户选择退出该窗口时，程序将会直接退出程序。这里只调用了两个 Unload 事件，分别是卸载本窗口和选择工作表窗口。在卸载选择工作表窗口中包含了退出系统的操作，该部分代码请见选择工作表窗口代码介绍。
- ListView 控件项目单击事件：因为可能该工作簿的路径字符串很长，造成该字符串无法被 ListView 控件全部显示。当 ListView 控件的项目被单击时，需要将被选定项目的路径信息通过 ToolTip 属性显示出来。

以下是这几个事件的代码解释：

```
Private m_fileLoc As String

Private Sub ListViewResult_ItemClick(ByVal Item As MSComctlLib.ListItem)
Item.TooltipText = Item.SubItems(1)              '设置提示信息内容
End Sub

Private Sub UserForm_Activate ()
刷新窗体语言显示 Me                               '根据用户语言显示设置刷新窗口语言显示
刷新已选工作表列表                                '刷新 ListView 控件的显示
刷新已选择表                                      '从数据库中载入 ListView 控件的项目
默认保存文件名                                    '初始化备选文件名复合框项目
End Sub

Private Sub UserForm_Terminate()
Unload Me                                         '卸载本窗口
Unload frm 选择工作表                             '卸载选择工作表窗口
End Sub
```

12.7.3 按钮代码设计

本窗口中包含了 3 个按钮，本小节将依次介绍这 3 个按钮控件的代码。以下是这 3 个按钮的功能介绍：

- 【浏览】按钮单击事件：单击该按钮后，弹出文件名路径获取窗口。然后程序将该路径字符串显示到路径文本框中，并且将该字符串保存到临时变量中，以供其他过程调用。
- 【上一步】按钮单击事件：单击该按钮后，将退回到选择工作表窗口。当用户需要修改备份工作表时，不能在该窗口中完成修改设置，必须回到选择工作表窗口中。
- 【确认】按钮单击事件：单击该按钮后，将调用合并工作表过程完成备份工作表任务。

以下是这几个按钮单击事件代码的解释：

```
Private Sub btnBrowse_Click()
Dim strFileLoc As String, defFileName As String
Dim strFileFilter As String
'设置文件筛选条件字符串
strFileFilter = "Excel2000-2003 file(*.xls),*.xls,Excel 2007 file(*.xlsx),*.xlsx"
If Len(ListFileName.Text) Then                         '检测用户是否选择了备选文件名
    defFileName = ListFileName & "-" & FormatDateTime(Now, vbShortDate) & _
                 "-" & Replace(FormatDateTime(Now, vbShortTime), ":", "")  '设置默认保存文件名
    '打开文件路径获取窗口，此时显示的文件名为默认文件名。最后将用户选择文件的路径保存到临时变量中
    strFileLoc = Application.GetSaveAsFilename(InitialFileName:=defFileName,
filefilter:=strFileFilter)
Else
    '打开文件路径获取窗口，将用户选择文件的路径保存到临时变量中
    strFileLoc = Application.GetSaveAsFilename(InitialFileName:="", filefilter:=strFileFilter)
End If
If InStr(1, strFileLoc, Application.PathSeparator) Then     '检测用户是否选择了文件
    m_fileLoc = strFileLoc                                 '保存备份文件名的路径
    txtFileLoc.Value = strFileLoc                          '将路径显示在文本框中
End If
End Sub

Private Sub btnOKLoc_Click()
If Len(txtFileLoc.Text) Then                             '检测用户是否设置了保存文件路径
    CombineWorkBook Trim(txtFileLoc.Text)                '开始备份工作表
End If
End Sub

Private Sub btnPreStepLoc_Click()
Me.Hide                                                  '隐藏本窗口
frm 选择工作表.Show                                       '显示选择工作表窗口
End Sub
```

代码说明：

在浏览按钮的单击事件代码中，最后确认用户是否选择了文件时，其判断的依据是最后的文件路径中是否包含了“\”字符。无论用户选择的文件位于根目录还是其他位置，都必然包含了路径分隔符号。而未选择文件时，返回的值一定不包含该符号。

12.7.4　刷新已选工作表列表过程代码设计

刷新已选工作表列表过程用于刷新 ListView 控件的显示。该过程同先前各个窗口中 ListView 控件的刷新过程类似。以下是该过程的代码解释：

```
Private Sub 刷新已选工作表列表()
With ListViewResult
    .ListItems.Clear                                        '清空控件所有项目
    .Gridlines = True                                       '显示网格线
    .FullRowSelect = True                                   '允许整行选择
    .MultiSelect = True                                     '允许多行选择
    .LabelEdit = lvwManual                                  '当单击项目时，不进入编辑状态
    .View = lvwReport                                       '设置控件显示模式
    With .ColumnHeaders
        .Clear                                              '清除标题
        If languageset Then                                 '检测语言设置项目
            .Add Text:="SheetName", Width:=Me.Width * 0.15  '设置英文表名列标题
            .Add Text:="BookPath", Width:=Me.Width * 0.7    '设置英文工作簿路径列标题
        Else
            .Add Text:="表名", Width:=Me.Width * 0.15       '设置中文表名列标题
            .Add Text:="所在工作簿路径", Width:=Me.Width * 0.7 '设置中文工作簿路径列标题
        End If
    End With
End With
End Sub
```

12.7.5　刷新已选择表过程代码设计

刷新已选择表过程代码用于从数据库中读取所有已经被确认需要备份的工作表记录到 ListView 控件中。程序首先从数据库中获取到工作表的记录集并将 ListView 控件所有项目清除，然后循环所有记录集项目，将记录集中所有记录的工作表名字段和工作簿路径分别写入 ListView 控件的对应列中。如图 12-40 所示的是该过程的流程图。

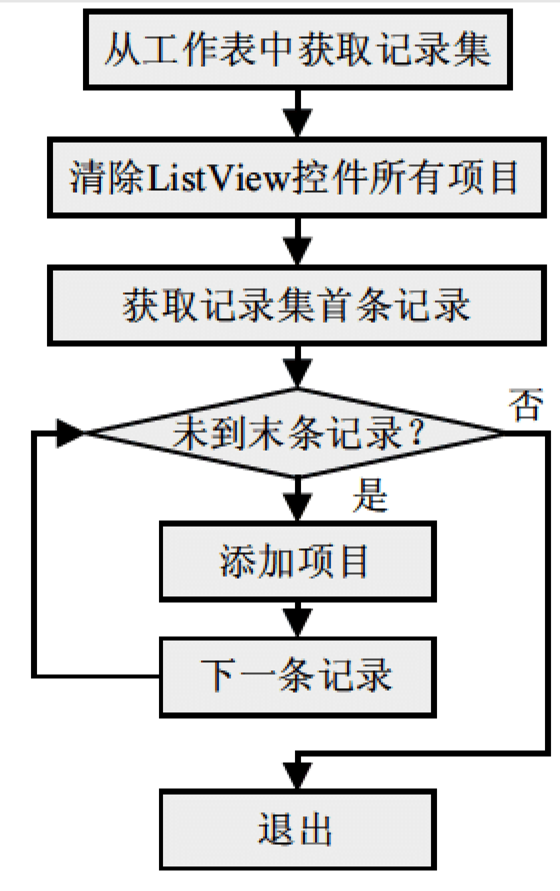


图 12-40　刷新已选择工作表过程流程图

以下是该过程的代码解释：

```
Private Sub 刷新已选择表()
On Error Resume Next
rs.Close                                          '关闭记录集
rs.Open "select * from 工作表", cnn 临时数据簿    '从工作表中获取记录集
ListViewResult.ListItems.Clear                    '清除 ListView 控件所有项目
If rs.RecordCount Then                            '检测记录集是否有记录
    rs.MoveFirst                                  '将记录集指针移动到第一条
    Do Until rs.EOF                               '循环直到记录集末端
```

```
            '添加项目
            With ListViewResult.ListItems.Add(Text:=Left(rs.Fields("工作表名"), Len(rs.Fields("工作表名")) -1))
                .SubItems(1) = rs.Fields("工作簿路径")             '设置工作簿路径
            End With
            rs.MoveNext                                           '移动记录集指针到下一条
        Loop
End If
On Error GoTo 0
Set rs = Nothing
End Sub
```

12.7.6 默认保存文件名过程代码设计

默认保存文件名过程用于从用户选择的所有工作簿中获取非重复工作簿名，然后将这些名称作为备选文件名复合框的项目添加，以供用户选择。

程序首先从数据库的工作表中获取非一致工作簿路径记录集，然后循环记录集中所有记录，从工作簿路径中获取工作簿名称。这些获得工作簿名称可能是重复的，在这里程序将剔除那些重复的工作簿名，然后程序将这些工作簿名称保存到一个集合对象中，最后程序将集合对象中获取的所有工作簿名都添加到备选文件名复合框中。如图 12-41 所示的是该过程的流程图。

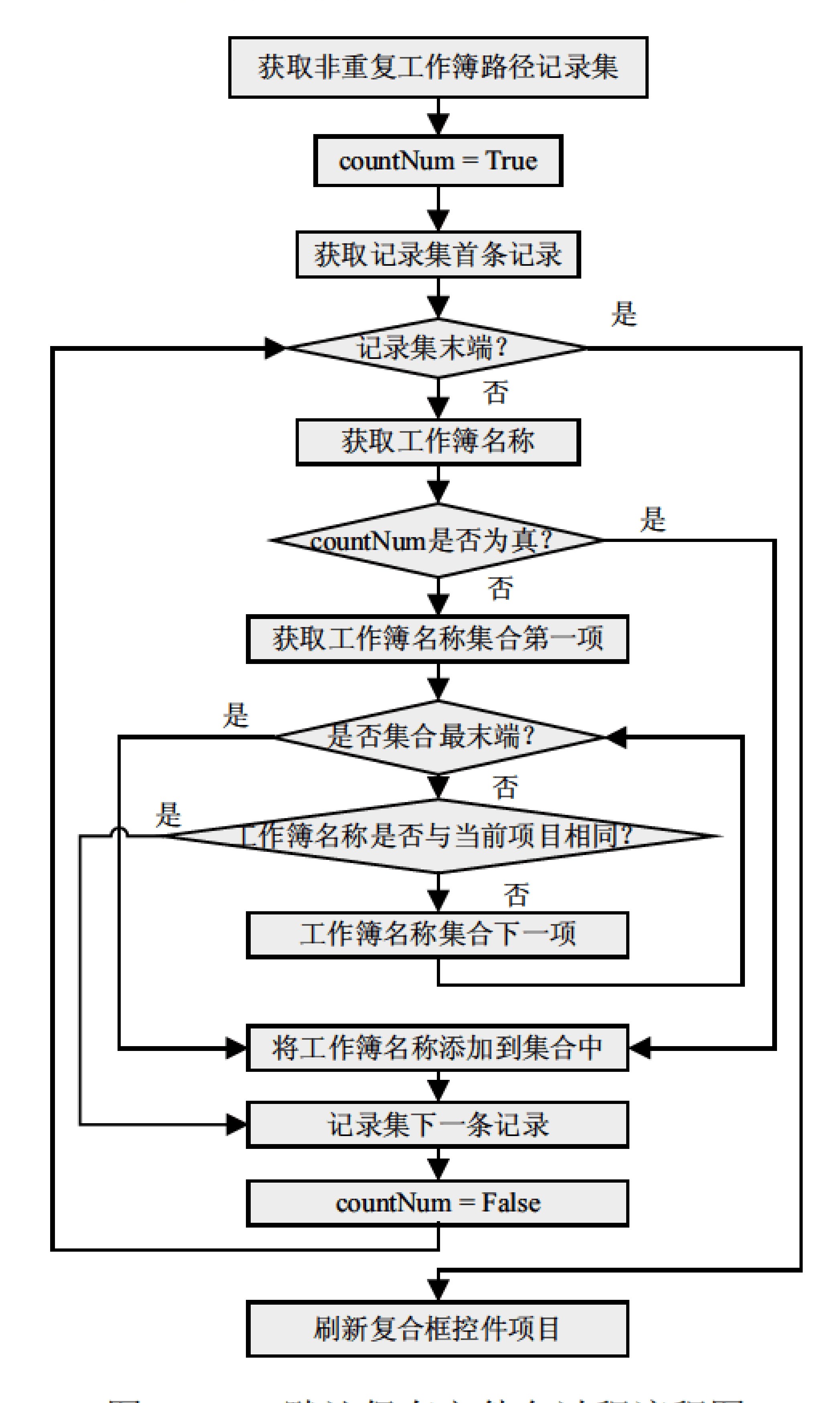


图 12-41　默认保存文件名过程流程图

在过程中，使用了一个 GoTo 语句，因而在上面的结构图中出现了一个穿越其他流程线的流程。读者可以对比下面的代码理解图 12-41。以下是该过程的代码解释：

```
Private Sub 默认保存文件名()
Dim colFileName As New Collection
Dim strfileName As String, intPos As Integer, countNum As Boolean
rs.Open "select distinct 工作簿路径 from 工作表", cnn 临时数据簿   '获取非重复工作簿路径记录集
countNum = True                                                  '初始化 countNum 变量
If rs.RecordCount Then                                           '检测记录集是否有记录
    Do Until rs.EOF                                              '循环直到记录集末端
        strfileName = rs.Fields("工作簿路径")                    '获取工作簿路径字符串
        If InStr(1, strfileName, Application.PathSeparator) Then '检测路径是否包含“\”字符
            获取工作簿名 strfileName                             '获取包含后缀的工作簿名称
        End If
        If InStr(1, strfileName, ".") Then                       '检测文件名是否包含点号
            strfileName = Left(strfileName, InStr(1, strfileName, ".") -1)
                                                   '获取去掉后缀与点号的工作簿名
        End If
        If countNum Then                           '检测 countNum 是否为真
            colFileName.Add strfileName            '将工作簿名直接添加到集合
        Else
            For i = 1 To colFileName.Count         '循环集合中所有工作簿名称
                If strfileName = colFileName(i) Then   '检测当前文件名是否重复
                    GoTo NextRS_Handle             ‘当重复时，跳到 NextRs_Handle
                End If
            Next
            colFileName.Add strfileName            '当没发现重复时，将该名称添加到集合
        End If
NextRS_Handle:
        rs.MoveNext                                '将记录移动到下一条
        countNum = False                           '标记 countNum 变量为假
    Loop
    ListFileName.Clear                             '清空复合框控件项目
    For i = 1 To colFileName.Count                 '循环集合所有项目
        ListFileName.AddItem colFileName(i)        '为复合框控件添加项目
    Next
    ListFileName.Text = colFileName(1)             '设置复合框默认显示值
End If
End Sub
```

12.8　信息提示窗口设计

在用户选择了保存工作簿位置并单击【确认】按钮后，可能用户设置的文件已经存在，此时需要提示需要询问用户下一步的操作方式。该信息提示窗口即用于提示用户文件已存在的消息，并且要求用户选择一种执行方式。该窗口中可以选择覆盖和添加，覆盖操作将删除已存在文件后，将所有需备份工作表添加到新工作簿中；添加操作将打开该工作簿，然后将

需备份工作表添加到该工作簿中。

12.8.1 窗口界面设计

该窗口的界面比较简单，包含了 1 个标签控件和 3 个按钮控件。这里不再加以介绍详细的设计步骤，只对控件加以文字说明。LabelTipInf 标签控件用于显示提示信息。【覆盖】与【添加】按钮被单击时，程序将把用户的选择保存到公共变量中。保存备份工作簿时程序根据该公共变量的值完成覆盖或添加操作。如图 12-42 所示的是该窗口的界面图。



图 12-42 信息提示窗口界面

12.8.2 窗口代码设计

窗口的代码不多，这里不再加以分节介绍。窗口代码包含了 4 个过程，分别是 ShowFormTipInf 自定义过程以及 3 个按钮单击事件过程。该窗口并不包含窗口初始化与激活事件。在保存工作簿过程中使用该窗口时，将该窗口看作了一个对象。通过调用该窗口对象的 ShowFormTipInf 过程完成初始化任务。以下是该窗口的代码解释：

```
Public Sub ShowFormTipInf(Language As Boolean)
With Me
    If Language Then                                          '检测语言设置变量
        .Caption = "File Exist already"                       '设置窗口标题
        .LabelTipInf.Caption = "File Exist aready! Please select operation!"
                                                              '设置标签控件 Caption 属性
        .btnReWrite.Caption = "ReWrite"                       '设置覆盖按钮的 Caption 属性
        .btnAddIn.Caption = "Add In"                          '设置添加按钮的 Caption 属性
        .btnCancel.Caption = "Cancel"                         '设置取消按钮的 Caption 属性
    Else
        .Caption = "文件已存在"                                '设置窗口标题
        .LabelTipInf.Caption = "文件已存在！  请选择操作！  "   '设置标签控件 Caption 属性
        .btnReWrite.Caption = "覆盖"                           '设置覆盖按钮的 Caption 属性
        .btnAddIn.Caption = "添加"                             '设置添加按钮的 Caption 属性
        .btnCancel.Caption = "取消"                            '设置取消按钮的 Caption 属性
    End If
    .Show
End With
End Sub

Private Sub btnReWrite_Click()
BtnClickIndex = 1                                             '标记用户单击了覆盖按钮
Unload Me                                                     '卸载窗口
End Sub

Private Sub btnAddIn_Click()
BtnClickIndex = 2                                             '标记用户单击了添加按钮
Unload Me                                                     '卸载窗口
```

```
End Sub

Private Sub btnCancel_Click()
BtnClickIndex = 3                                        '标记用户单击了取消按钮
Unload Me                                                '卸载窗口
End Sub
```

12.9　系 统 测 试

本系统包含了两个功能模块，在该节系统测试部分也将分为两个部分分别介绍这两个功能模块的运行流程。需注意的是，本系统处理的文件只包含普通数据的 Excel 文件，因而读者在测试该加载宏时，请不要使用其他文件类型。

12.9.1　拆分工作簿模块功能测试

该小节测试部分使用的实例可以在光盘的测试实例中找到。该小节使用的工作簿文件是“拆分工作簿.xls”。由于系统将新拆分工作簿保存在该文件同一目录下，在读者测试时请将该文件复制到硬盘中，以免发生程序向光盘写入数据的错误。该文件包含了 3 个工作表，测试中将把这 3 个工作表分开保存到 2 个工作簿中。第二个和第三个工作表被保存在同一个新工作簿中。以下是测试过程：

（1）打开本加载宏文件后，在 Excel 2007 菜单中依次选择【加载项】|【工作簿拆分与备份】|【拆分工作簿】命令。随后程序打开拆分工作簿设置窗口。单击窗口中的【浏览】按钮，找到“拆分工作簿.xls”文件后确认。此时未分配表列表中将刷新出该工作簿的所有工作表。其效果如图 12-43 所示。



图 12-43　测试拆分工作簿

（2）在已分配工作表复合框中输入分组，这个分组将作为新工作簿的名称，这里无需写入后缀名称。在这里输入“拆分簿 1”，然后单击【添加】按钮。添加成功后，会出现一个提示添加成功的消息框（如图 12-44 所示）。接着再添加一个分组“拆分簿 2”，添加完成后的效果如图 12-45 所示。



图 12-44　提示分组项目添加成功



图 12-45　添加分组项目

（3）确认分组为拆分簿 1，然后在未分配工作表列表中选中【拆分表 1】复选框，随后单击【添加】按钮（如图 12-46 所示）。此时该拆分表将从未分配表列表中删除，而被添加到了已分配工作表列表中（如图 12-47 所示）。



图 12-46　选中未分配表并添加



图 12-47　分配工作表

（4）在已分配工作表分组复合框中选择“拆分簿 2”，然后依照前面的操作，将“拆分表 2”、“拆分表 3”添加到“拆分簿 2”分组中。此时未分配表列表中已经没有项目，说明已经分配完毕。这时的设置效果图如图 12-48 所示。单击【开始拆分】按钮后，程序将按照以上设置拆分工作簿的工作表。拆分完成后获得两个新工作簿文件，如图 12-49 所示。



图 12-48　分配剩余工作表



图 12-49　拆分结果工作簿

12.9.2　备份工作簿模块功能测试

备份工作簿的测试也使用到了测试文件。在上面文件同一目录下，读者可以找到两个 Excel 2007 文件“合并工作簿 1.xlsx”和“合并工作簿 2.xlsx”。它们各自包含 3 个工作表，以下的测试将从第一个工作簿中取出一个工作表，而从第二个工作簿中也取出一个工作表，然后将这两个工作表备份到一个新工作簿中。在开始测试之前，请读者也将这两个文件复制到硬盘中。

（1）在 Excel 2007 菜单中依次选择【加载项】|【工作簿拆分与备份】|【备份工作簿】命令，随后程序打开备份工作簿设置窗口。在该窗口中单击【打开】按钮，在打开的文件获取窗口中找到这两个文件，一次性打开即可（如图 12-50 所示）。此时这两个工作簿的信息被自动刷新到列表中。然后选择【全部勾选】复选框，将两个工作簿选中。这时【下一步】按钮被激活，单击【下一步】按钮进入下一步设置，效果如图 12-51 所示。

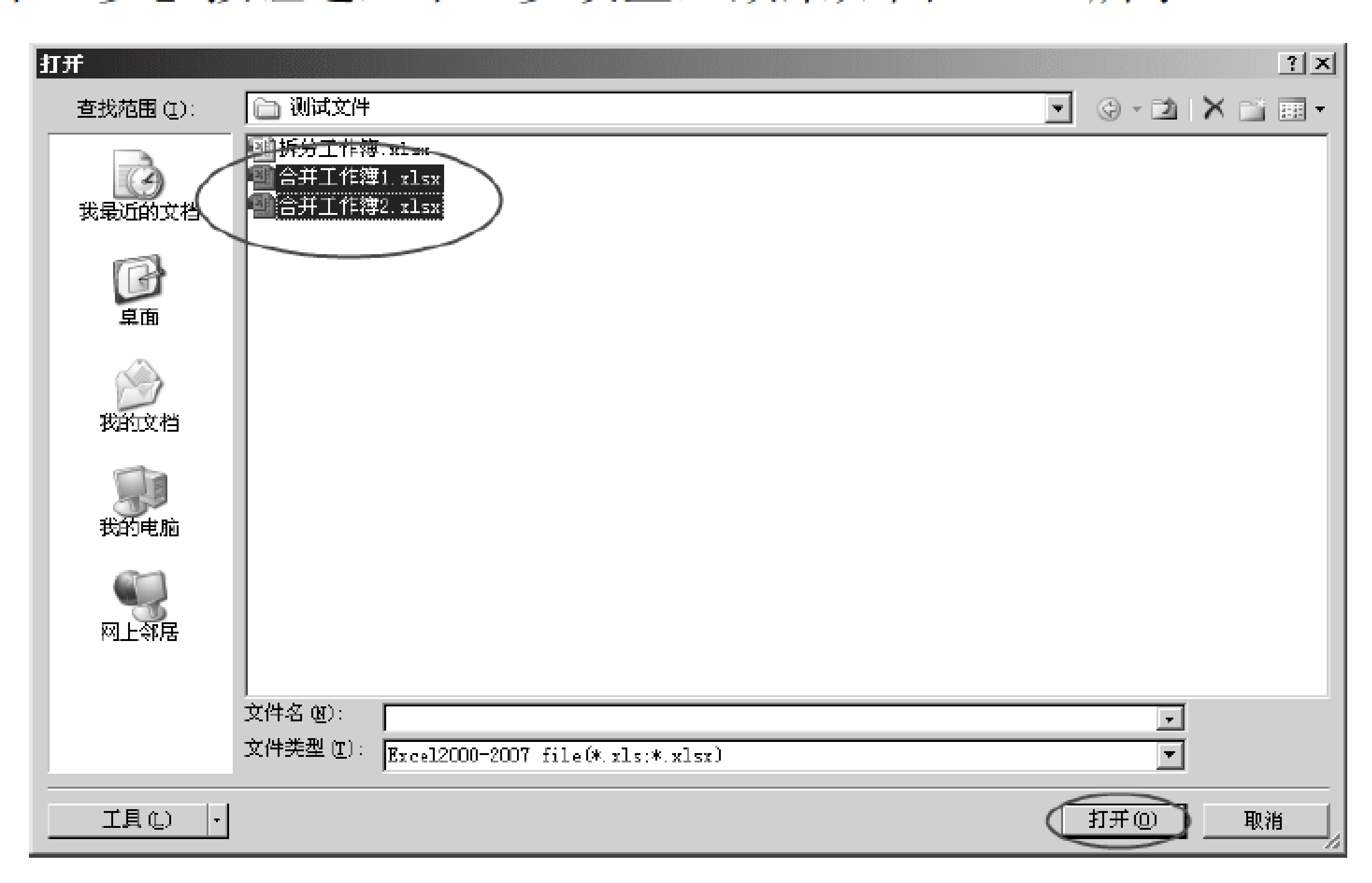


图 12-50　工作簿获取窗口



图 12-51　选中工作簿并进入下一步设置

（2）在打开的工作表设置窗口的当前工作簿复合框中选中【合并工作簿 1】，然后在工作表列表中选择【合并表 1】（如图 12-52 所示），随后在当前工作簿复合框中选中【合并工

作簿 2】，并在工作表列表中选中【合并表 6】（如图 12-53 所示），然后单击【下一步】按钮进入保存工作簿位置设置步骤。

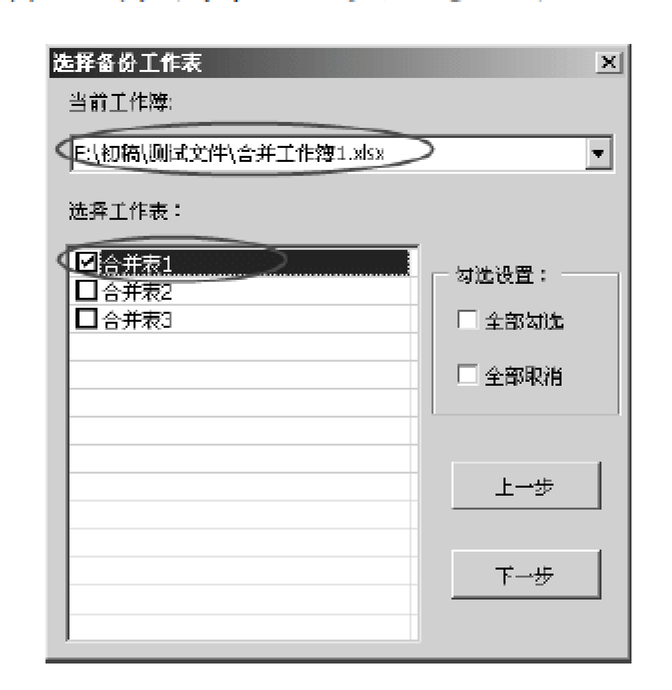


图 12-52　选择第一个工作簿的工作表

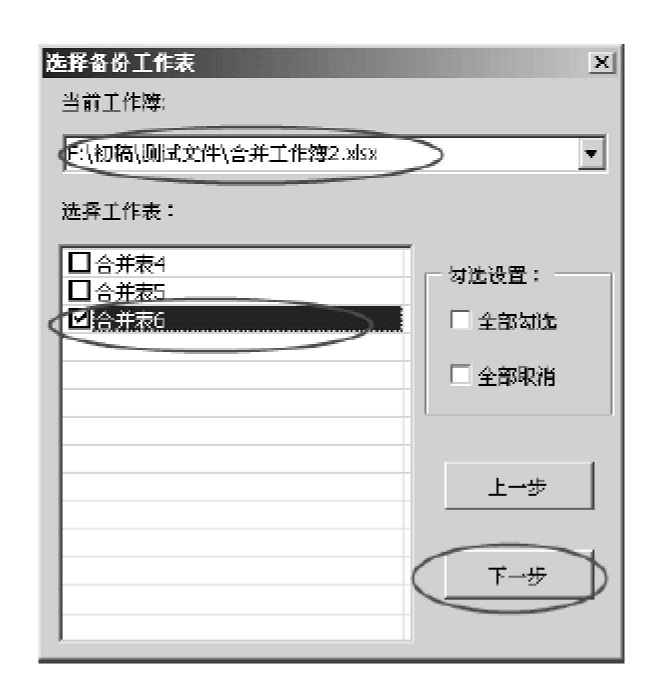


图 12-53　选择第二个工作簿的工作表

（3）在保存位置设置窗口中单击【浏览】按钮，此时程序已经根据默认文件名自动生成文件的名称。当然读者也可以自己修改该名称，这里就使用了这个默认名称，以防有重复文件存在（如图 12-54 所示）。设置好保存位置后，单击窗口中的【确定】按钮（如图 12-55 所示）。程序将按照前面的各步设置备份工作表到新工作簿中。备份完成后的结果如图 12-56 所示。

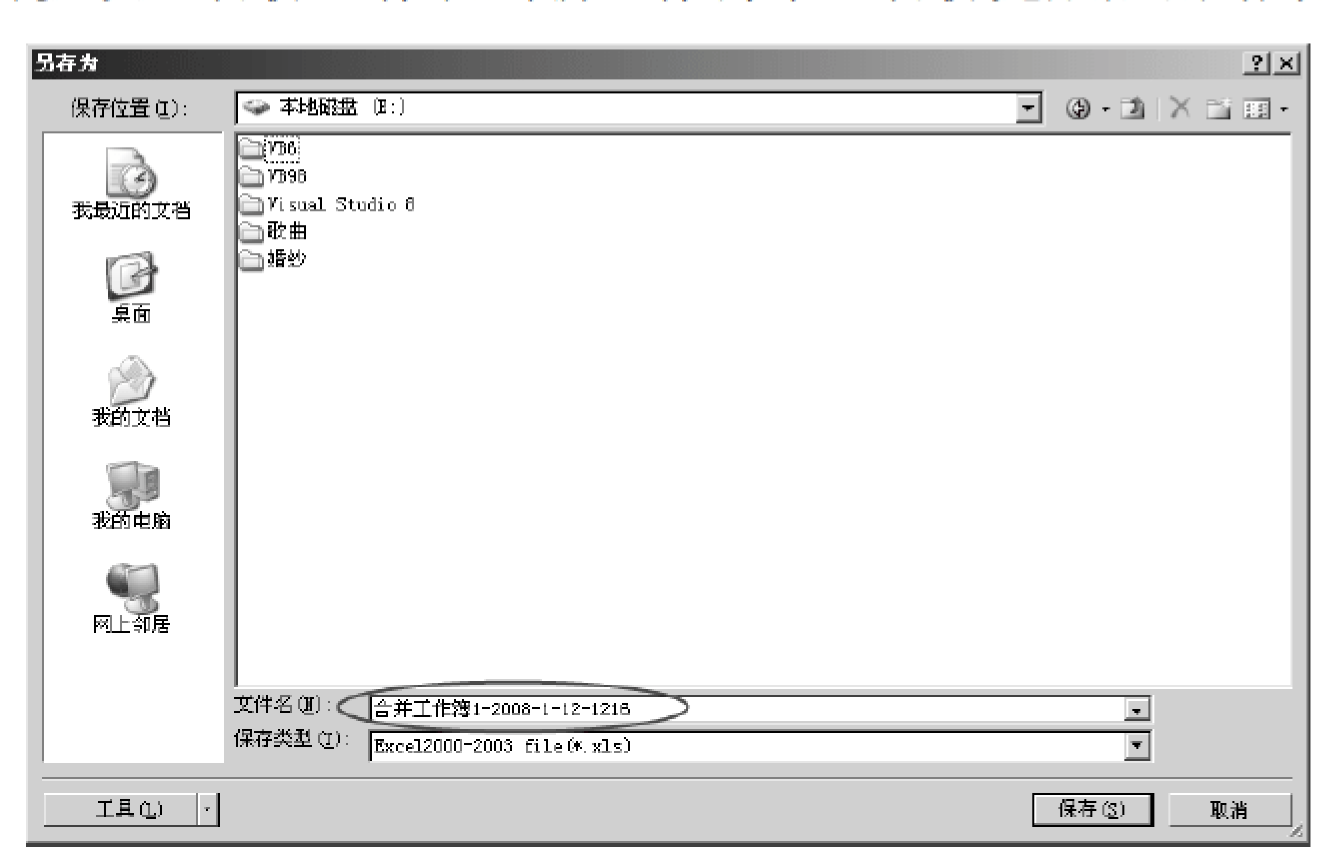


图 12-54　保存工作簿文件位置

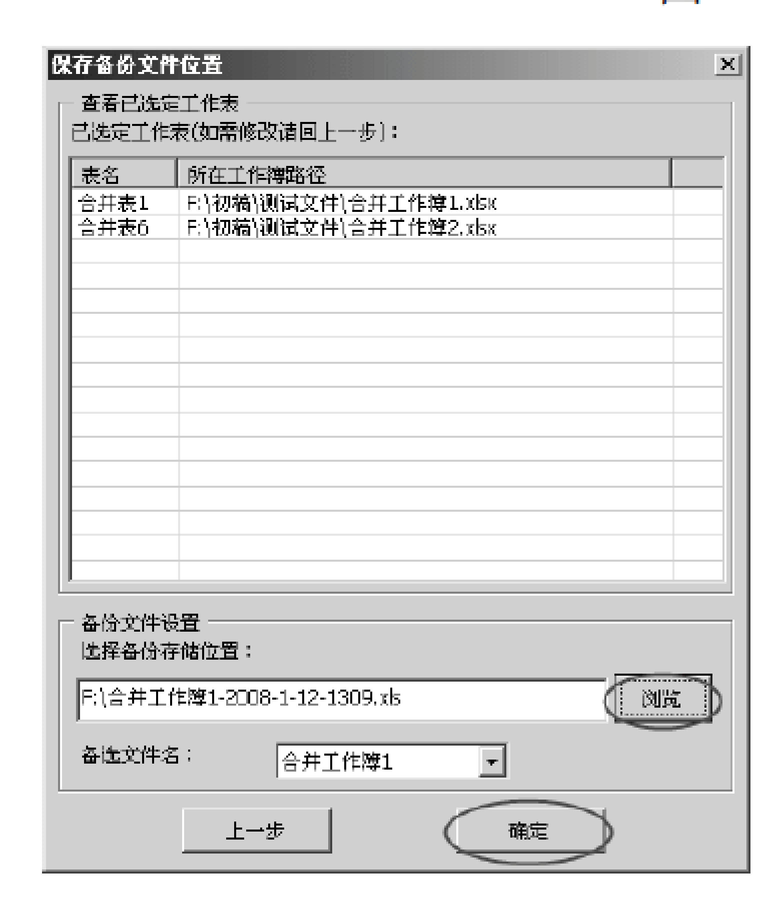


图 12-55　设置保存位置并开始备份

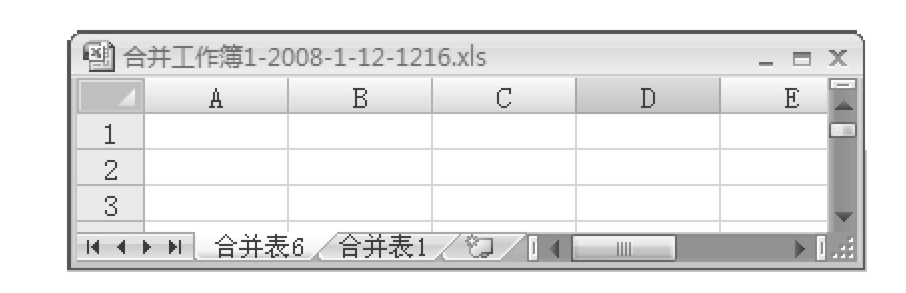


图 12-56　备份工作簿文件

读者意见反馈卡

您购买的书名：________ 您的姓名：________ 性 别：□男 □女
年龄：________ 文化程度：________ 职 业：________
邮编：________ 通信地址：________ E-mail：________
您常用的软件：1 ________ 2 ________ 3 ________ 4 ________

您购买本书的原因（可多选）：

□封面与装帧 □引言目录 □正文内容 □丛书风格 □价格 □光盘 □专业性强 □别人介绍
□出版社或作者名声 □售后服务

本书最令您满意的是（可多选）：

□专业性强、覆盖面广 □内容翔实、定位准确 □精益求精、售后服务

您可以承受的图书价格：

□20 元以下 □30 元以下 □40 元以下 □50 元以下 □只要内容好，不论价格

您对本书的评价：

封面装帧：□很好 □较好 □一般 □不满意 建议________
印刷质量：□很好 □较好 □一般 □不满意 建议________
正文质量：□很好 □较好 □一般 □不满意 建议________
写作风格：□很好 □较好 □一般 □不满意 建议________
专业水平：□很好 □较好 □一般 □不满意 建议________

您希望增加哪些图书选题：1________ 2________ 3________

您认为本书有哪些错误：

章____节____页码____行____列____ 图号____ 错误______应改为 ______
章____节____页码____行____列____ 图号____ 错误______应改为 ______
章____节____页码____行____列____ 图号____ 错误______应改为 ______
章____节____页码____行____列____ 图号____ 错误______应改为 ______

您的其他建议：

1________
2________
3________

请填好本卡后寄给：

清华大学校内金地公司 邮编：100084 电话：（010-82899771）
《Excel VBA 应用开发经典案例》编辑部收
传真：（010）62788903 公司网址：www.thjd.com.cn E-mail：zhuyingbiao@126.com
如需本书可与本编辑部联系邮购，汇款请按以上地址填写，另加邮费 15%（挂号）